I MANUALI DI BASIC
20 PROGRAMMI PER
COMMODORE 64
• Poker d'assi
• Un medico poco serio
• Computo, computas...
• Strike and ball
• La torre di Hanoi
• Grafica monodimensionale
• Corsa d'auto
• Alta risoluzione
• Disegna il tuo sprite
• Grafica tridimensionale
• Slot machine
• Proviamo con l'Assembler
• Labirinto
• Il gioco della vita
• Memory dump
• Archivio
• Aritmetica a precisione infinita
• L'archivio veloce
• Editor
• Le otto regine
ARMANDO CURCIO EDITORE
IN EDICOLA
uno splendido volume
di 192 pagine stampate a colori
su carta patinata speciale
con copertina plastificata,
rilegato a spirale
per un uso pratico e facile
20 programmi
a sole L. 8.000
Nei prossimi volumi
nuovi programmi per altri
personal e home computer
attualmente presenti
sul mercato

Sommario

RUBRICHE

Redazione/collaboratori: Giovanni Bellù, Andrea e Alberto Boriani, Giancarlo Castagna, Eugenio Coppari, Marco De Martino, Giancarlo Mariani, Enrico Scelsa, Fabio Sorgato, Danilo Toma
Segreteria di redazione: Maura Ceccaroli, Piera Perin
Impaginazione/illustrazioni: Francesco Amatori, Renato Caruso
Direzione, redazione, pubblicità: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Mirco Croce (coordinatore), Giuseppe Porzani, Michela Prandini, Giorgio Ruffoni, Claudio Tidone, Villa Claudio - Segretaria: Liliana Degiorgi
● Roma: Spazio Nuovo - via P. Foscari 70 - 00139 Roma - Tel. 06/8209679
Abbonamenti: Marina Vantini
Tariffe: prezzo per copia L. 3.000. Abbonamento annuo (11 fascicoli) L. 28.000. Estero: il doppio. Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 55.000.
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizioni: Systems Editoriale Srl
Fotolito: Systems Editoriale Srl
Stampa: La Litografica S.r.l. - Busto Arsizio (VA)
Registrazione: Tribunale di Milano n. 370 del 2/1/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70%
Distribuzione: MePe, via G. Carcano 32 - Milano

Da questo mese la rivista C.C.C. è disponibile anche su nastro

Precisazioni

Per una serie di circostanze sfavorevoli gli articoli di Danilo Toma pubblicati sui n. 16 (pag. 18—22) e 17 (pag. 69) risultano in parte imprecisi.

□ Nella colonna centrale di pag. 18 (ccc n.16) vi è un riferimento alle linee 1000—1800 che dovrebbero contenere nomi e indirizzi dei comandi grafici (pubblicati nel n. 14) mentre tali linee non compaiono nel listato pubblicato.

La conoscenza di tali indirizzi è importante soprattutto dopo la pubblicazione dell'articolo riguardante i comandi per registrare la pagina grafica (C.C.C. n. 17).

```
1000 PLOT=50701
1100 DRAW=50720
1200 CIRCLE=50745
1300 ARC=50754
1400 CLEAR=49763
1500 GRAF=50858
1600 TEXT=50890
1700 MGRAF=50922
1800 COLOR=50986
```

□ Nel paragrafo "ELIMINAZIONE DELLA FRECCETTA" (pag. 19 N.16) viene fatto riferimento a numeri di linea delle "routines grafiche II". Purtroppo le linee citate non corrispondono a quelle del listato pubblicato nel numero 14.

Nella frase "...cambiate il primo DATA della linea 24250...", tale numero di linea va sostituito con 2250. Tre righe più avanti, al posto di 24300, va messo 2260.

Infine, due righe dopo, il numero giusto è 225 invece di 17.

□ Il paragrafo "UNA APPLICAZIONE" non doveva comparire nell'articolo in quanto (vedi n. 17 di C.C.C.) era stato "scorporato" e sviluppato autonomamente, sfociando nell'articolo sulla registrazione della pagina grafica.

□ Nel paragrafo "UN ESEMPIO" si fa riferimento a due figure mentre bisognava parlare di due listati.

In realtà quando trovate "figura 1" dovete intendere "listato lungo" mentre "figura 2" equivale a "listato breve".

Quanto detto dovrebbe aver chiarito ogni dubbio. Una preghiera: non scrivete a Danilo Toma per avere informazioni, consigli o altro perchè non ha il tempo per rispondere se non attraverso l'apposito tagliando che trovate in fondo alla rivista.

Eccesso d'automatismo

□ Quando digito SAVE, oppure LOAD, e premo il tasto Return, non appare il messaggio PRESS PLAY ON TAPE ma il quadro scompare subito come se il tasto fosse già premuto. I programmi vengono caricati, salvati e verificati normalmente. Perchè questo strano comportamento? *(Antonio Perrone - Taranto)*

• Il registratore possiede, al suo interno, un interruttore che viene chiuso quando si preme uno dei tasti play, record, rewind eccetera. Il computer, nei casi di registrazione o lettura, attende che tale interruttore sia chiuso per iniziare le operazioni col nastro.

E' quindi molto probabile che tale interruttore sia chiuso permanentemente per un difetto meccanico oppure per un cortocircuito responsabile dell'invio del falso messaggio "tasto premuto!".

E' anche probabile che il connettore posto sul resto del computer sia in corto circuito. Per sincerartene, prova ad utilizzare il registratore di un tuo amico. Se l'inconveniente scompare vuol dire che il registratore è in corto. In caso contrario il difetto risiede nel computer. Come, però, giustamente fai notare, l'inconveniente è minimo e non pregiudica il corretto funzionamento del calcolatore.

Checksum

□ In che cosa consiste il checksum nei listati? *(Massimo Davi - Ostiglia)*

• Quando in un programma pubblicato su di un libro si fa ricorso ad istruzioni del tipo READ...DATA è molto facile incorrere in errori di trascrizione da parte del lettore. Nei casi, ad esempio, di programmi in Linguaggio Macchina vi sono decine di lavori numerici DATA che, letti uno ad uno, vengono allocati in memoria mediante istruzioni POKE. Nel Linguaggio Macchina, come forse saprai, è sufficiente sbagliare a digitare un solo valore per "inchiodare" il computer. Ciò significa che è necessario spegnerlo e riaccenderlo perdendo, di conseguenza, tutto il contenuto della memoria.

Per evitare tale inconveniente, gli autori di programmi pubblicati sulle riviste

(segue a pag. 70)

PASCALINA E ARITMOMETRO

All'inizio fu il numero. Era il 1642. Blaise Pascal, francese diciannovenne, filosofo, scrittore e matematico pensò una macchina per fare operazioni aritmetiche. Una serie di ruote collegate da ingranaggi rappresentavano le decine, le centinaia, le migliaia. Su ogni circonferenza erano poi calcolate le cifre, dallo 0 al 9. Azionando una ruota, avanzava di un'unità anche la ruota alla sua sinistra.

Si chiamava Pascalina, e per l'umanità intera rappresentava un buon primo passo sulla strada dell'elaborazione dei dati. Certo, i calcoli andavano ancora impostati manualmente. Ma il mutamento era concettuale, faceva parte del mondo delle idee. Per la prima volta era dimostrato che il calcolo poteva essere automatizzato: una vera rivoluzione. E, fuori dal mondo delle idee per entrare a far parte della storia del mondo delle cose, per la prima volta una macchina eseguiva automaticamente il riporto, secondo un meccanismo che sarebbe rimasto immutato per più di tre secoli. Idee forti, tecnologie primitive.

Bisognerà aspettare la rivoluzione (e, più ancora, la meccanizzazione del nostro secolo) perchè le istituzioni dei primi matematici potessero raggiungere tutti. Allora, si trattava più che altro di passatempi di intellettuali geniali.

E, curiosamente, filosofi. Leibniz, le monadi... a scuola viene studiato sui libri di filosofia. Ma anche lui entra a pieno merito nella storia, che è poi quella che porta ai computer che usiamo. Con un sistema di pignoni dentati, nel 1671 fu proprio Gottfried Wilhelm Leibniz a perfezionare la macchina di Pascal e ad aggiungervi moltiplicazioni e divisioni (sotto forma di addizioni e sottrazioni ripetute).

Più pratico ancora (ma il salto è di 200 anni) era l'aritometro, un congegno inventato dal francese Thomas de Colmar (fu anche il primo grande successo di vendite per una macchina calcolatrice: alcune migliaia di esemplari venduti dal 1820 al 1890).

Sotto, *interno della Pascalina, la macchina inventata da Blaise Pascal* **(nel riquadro)** *in grado di eseguire addizioni e sottrazioni, nonchè il riporto automatico. Filosofo e scrittore, Pascal inventò la macchina a soli 19 anni, per aiutare il padre, esattore delle tasse a Rouen.* **A sinistra,** *ritratto di un altro grande filosofo, il tedesco Gottfried Wilhelm Leibnizm. A lui si deve l'intuizione delle possibilità d'uso del sistema binario nelle macchine da calcolo.*

E poco tempo dopo, nel 1832, arriva la prima macchina analitica. A progettarla (non venne mai costruita) fu Charles Babbage, utilizzando la tecnica di riporto e uno strano aggeggio inventato 30 anni prima, la scheda perforata. Nasce con lei il calcolatore, o per lo meno la sua concezione generale. E anche noi, con le immagini di questo fascicolo ci fermiamo qui, alle soglie del nostro secolo, alla prima applicazione di massa di elaboratore è scheda perforata nel censimento americano del 1890. 50 anni dopo arriverà il primo calcolatore elettronico vero e proprio (nel 1943, l'ENIAC progettato nell'università della Pennsylvania).

IN EDICOLA

TUTTO I
IN 4

L BASIC NASTRI

Le nuove Maxell

UD I

Vanta una tecnologia senza confronti nella sua categoria. Infatti tutte le proprietà magnetiche del suo nastro sono esaltate al massimo grazie alla nuova particella magnetica "Ferricrystal" che, a differenza di quelle tradizionali, è **assolutamente non porosa**. La UD I, quindi, offre un livello d'uscita migliorato su tutta la gamma di frequenze, specialmente nelle medie e basse; una più ampia gamma dinamica, ed ottime caratteristiche di lownoise. Adotta la meccanica P.A. (PHASE ACCURACY) per garantire la massima stabilità di svolgimento del nastro e per contenere la differenza di fase tra i canali stereo entro i 10°. Può essere usata su qualsiasi tipo di registratore.

UD II

È la soluzione ideale per chi vuole una cassetta di **categoria superiore** ad un **costo contenuto**. Utilizza un nastro di posizione "Chromo" (CrO_2) prodotto con la stessa tecnologia dei nastri XL e XL-S Maxell. Infatti il nastro della UD II è composto dalle collaudate particelle magnetiche "Fine Epitaxial", ulteriormente perfezionate. Di conseguenza è aumentata tutta la **gamma dinamica** del nastro, il livello d'uscita alle **medie** ed **alte** frequenze, mentre il livello di rumore di bias ed il rumore di modulazione sono ridotti praticamente a **zero**.

Il meccanismo di scorrimento P.A. e la perfezione dei gusci della cassetta assicurano un regolare svolgimento del nastro, mantenendolo sempre perpendicolare alla testina del registratore (differenza di fase entro i 10°).

È TUTTA UN'ALTRA MUSICA.

DELPI

GAME PAVLOV

Trasformarsi in un topo. Immedesimarsi in un animale da laboratorio per mettere alla prova il proprio istinto.

Nella storia della scienza lo studio del comportamento degli animali è sempre stato di fondamentale importanza. Gli esperimenti in questo campo, sono infatti moltissimi, e vanno dal campo della medicina allo studio delle facoltà mentali. Uno di questi esperimenti il cui precursore è il famoso PAVLOV, consisteva nel cercare di capire come funzionano i riflessi condizionati. L'animale analizzato in uno degli esperimenti era un grazioso topolino immesso, per l'occasione, in una speciale gabbia. Questa era divisa in due scompartimenti comunicanti, ed il topolino poteva decidere liberatamente se stare da una parte, dall'altra...o riposarsi.

Lo sperimentatore, però, non lo lasciava mai tranquillo: non appena il topolino si fermava, egli accendeva una lampadina per breve tempo, e, dopo un paio di secondi, una scarica elettrica fastidiosa (leggera, per fortuna del topolino), veniva diffusa sulla griglia metallica che formava il pavimento della gabbietta. A questo punto il topolino iniziava una corsa disperata e, con sua grande meraviglia, una volta arrivato nell'altro scompartimento non sentiva più quella noiosa scossa nelle zampe.

Avete 150 secondi per riuscire a capire la logica di questo gioco.

Lo sperimentatore passò alla seconda fase dell'esperimento, consistente nell'accendere lampadina e di mandare nuovamente una scarica nello scompartimento in cui era rifugiato il topolino. Dopo diverse prove il topolino riuscì a capire che se andava nell'altro scompartimento della gabbia appena vedeva accendersi la luce, poteva evitare la scossa, e così fece ogni volta che vedeva la luce accendersi. In seguito la luce fu accesa senza l'invio della scarica elettrica. Il risultato fu che il topolino continuava ugualmente a rifugiarsi dall'altra parte della gabbia.

L'azione del topolino, pur se volontaria era diventata un riflesso condizionato dalla lampada. Allo stesso modo se proviamo ad emettere un fischio ogni volta che diamo da mangiare ad un cane, dopo un po' di volte, limitandoci a fischiare, il cane si precipita felice e con l'acquolina in bocca. Agendo, cioè, esattamente come se dovesse mangiare.

Il programma

Lo scopo del programma che pubblichiamo è quello di proseguire lo studio dei riflessi condizionati utilizzando come cavia l'ideatore degli esperimenti prima descritti: l'uomo. E' una sfida. Obiettivo: vedere se anche noi riusciamo a capire la strategia migliore per cavarsela in condizioni sfavorevoli.

Il topolino è riuscito a capire che cosa doveva fare per non sentire la scossa: riu-

Pavlov e Master Mind

L'idea del programma pubblicato in queste pagine deriva dallo studio attento di un famoso programma, il Master Mind, che chiunque ha sicuramente digitato in una delle sue innumerevoli versioni. In un precedente editoriale avevo pregato i lettori-autori di non inviare più versioni del gioco, pur se elaborate.

Esaminando, comunque, l'elevato numero di listati pervenuti in redazione abbiamo deciso di intervenire suggerendo ciò che noi intendiamo per "sofisticazioni" di un listato.

L'omino che appare nel listato di Pavlov deve percorrere il labirinto "toccando" in successione due dei dieci simboli che appaiono lungo il percorso. A seconda dell'esito del contatto, il bordo del video cambia, o meno, del colore. Contemporaneamente il punteggio viene modificato, come pure il tempo a disposizione.

Con Pavlov, insomma, intendiamo proporre solo un'idea che, ce lo auguriamo, possa dare inizio al Master Mind della seconda generazione. Ad un gioco, cioè, che in base al comportamento del giocatore fornisca una risposta (meglio se... sibillina) da tenere in grande considerazione per il proseguimento del gioco stesso.

Per ciò che invece riguarda lo specifico gioco proposto ne avevamo, in un primo tempo, dato la descrizione completa. In seguito, però, è stata presa la decisione di non pubblicarla.

Il perchè di tale iniziativa è dovuto al fatto che, nella realtà, un topolino in una gabbietta non sa minimamente ciò che sta per capitargli ed anzi, credetemi, ne farebbe volentieri a meno.

sciremo anche noi in questo? Visto che l'uomo così si dice è più intelligente degli animali, i problemi da superare non dovrebbero essere poi insormontabili. Unica differenza (e grosso ostacolo): bisogna ragionare!

Il gioco

Immaginate di essere rinchiusi in un labirinto sotterraneo controllato, guarda caso, da un computer. Con il passare del tempo, sempre visualizzando sulla parte destra del video, il computer toglie dal labirinto una certa quantità di ossigeno. L'ossigeno tenderà cioè ad esaurirsi fino a determinare la conclusione del gioco.

Avete però la possibilità di salvarvi, o quanto meno di sopravvivere più a lungo usando una certa strategia. Lungo il labirinto, infatti, vi sono dei punti che, se collegati fra loro in modo esatto, permettono di azzerare il timer del computer centrale, con la seguente fornitura d'altro ossigeno aumentando, così, il punteggio.

Se però collegate tra loro punti non previsti, ecco che il tempo limite cambia (può essere in meglio o in peggio, non si sa però a priori). In ogni caso ancora a disposizione ancora del tempo (e quindi ossigeno) per cercare altre combinazioni. Potete usare una combinazione già scoperta, ma... Di più non posso dirvi, altrimenti il nostro esperimento non avrebbe motivo di esistere.

Cercate la strategia che permetta di realizzare il miglior punteggio e buona fortuna.

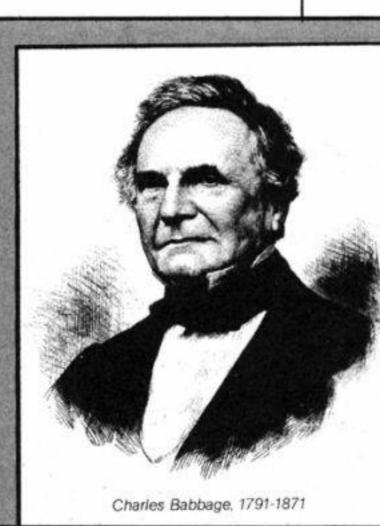
Giovanni Bellù

```
530 DEF  FND(A) = PEEK(1026+FNA(A
    ))
540 DEF  FNE(A) = PEEK(1064+FNA(A
    ))
550 DEF  FNF(A) = PEEK(1066+FNA(A
    ))
560 DEF  FNG(A) = PEEK(1104+FNA(A
    ))
570 DEF  FNH(A) = PEEK(1105+FNA(A
    ))
580 DEF  FNI(A) = PEEK(1106+FNA(A
    ))
590 DEF  FNL(A) = PEEK(1065+FNA(A
    ))
600 DEF  FNM(A) = INT(RND(1)*10)+
    1
610 REM
620 REM
630 X = 32 : Y = 58 : X1 = X : Y1
     = Y
640 V = 53248 : FOR  K = V  TO  V
    +40 : POKE  V,0 : NEXT
650 POKE  2040,13 : POKE  V+39,2
    : POKE  V+21,1
660 POKE  V,X : POKE  V+1,Y
670 POKE  V+23,0 : POKE  V+38,7
680 POKE  V+29,0 : OM = 13
690 POKE V+16,0 : POKE V+37,4
700 POKE V+28,1
710 REM
720 REM  MEMORIZZAZIONE DATI SPRI
    TE
730 REM
140 REM           COMMODORE 64
170 REM        "RIFLESSI CONDIZIONA
    TI"
220 REM
230 REM          DI GIOVANNI BELLU
    '
330 REM       ********************
    ***
340 REM       * JOYSTICK IN PORTA
```

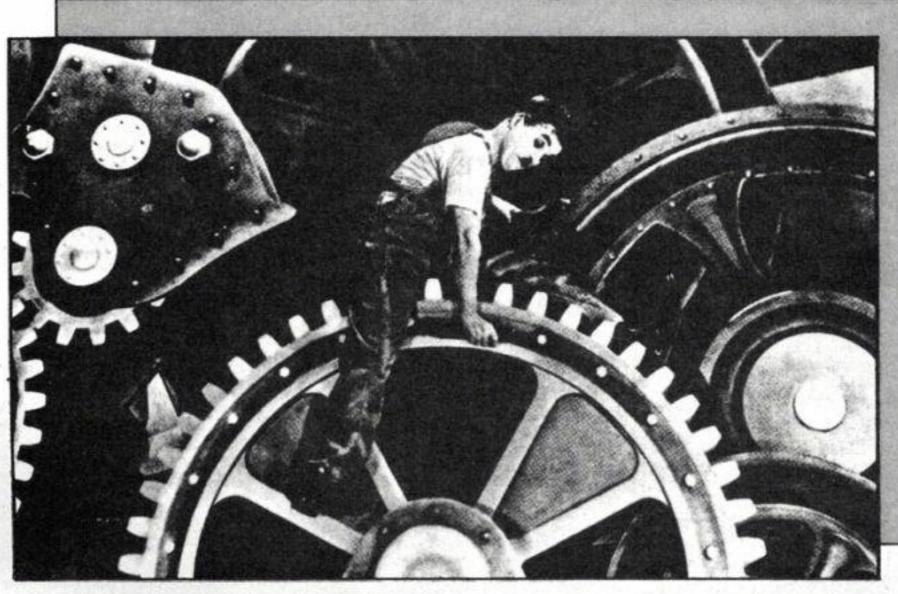
```
     1 *
 370 REM         ********************
     ***
 380 REM
 390 PRINT "[CLEAR][GIALLO]" :REM
      CLR/HOME E GIALLO
 400 REM  SFONDO NERO
 410 POKE  53280,0
 420 POKE  53281,0
 430 A = 1026
 440 DIM  A(10) : DIM  B$(10)
 450 DIM  C(10) : DIM  D (10)
 455 DIM  S(10,3)
 460 REM
 470 REM  DEFINIZIONE FUNZIONI
 480 REM  PER LA GESTIONE DEGLI SP
     RITE
 490 REM
 500 DEF  FNA(A) = (X-24)/8+((Y-50
     )/8)*40
 510 DEF  FNB(A) = PEEK(1024+FNA(A
     ))
 520 DEF  FNC(A) = PEEK(1025+FNA(A
     ))
 740 FOR  K = 0  TO  62
 750 READ  A : POKE  832+K,A : NEX
     T
 760 FOR  K = 0  TO  62
 770 READ  A : POKE  896+K,A : NEX
     T
 970 A = 1024
 980 REM
 990 REM  INIT. SUONO
1000 REM
1010 FOR  K = 54296  TO  54330 : P
     OKE  K,0 : NEXT
1020 POKE  54278,255 : POKE  54276
     ,17 : POKE  54296,15 : W = 54
     273
1030 REM
1040 REM  DATI PER RIFLESSI CONDIZ
     IONATI
1050 REM
1060 A(1)=81 : A(2)=87 : A(3)=42 :
      A(4)=83
1070 A(5)=88 : A(6)=90 : A(7)=65 :
      A(8)=36
1080 A(9)=35 : A(10)=43
1081 GOSUB  10000
1090 B$(1) = "000125" : B$(2) = "0
     00120"
1100 B$(3) = "000110" : B$(4) = "0
     00100"
1110 B$(5) = "000050" : B$(6) = "0
     00040"
1120 FOR  K = 0  TO  10  : C(K)  =
      K+2 : NEXT
1130 D(1) = 08 : D(2) = 09 : D(3)
     = 10
1140 D(4) = 11 : D(5) = 12 : D(6)
     = 14
1150 D(7) = 16 : D(8) = 17 : D(9)
     = 19
1160 D(10)= 21
1170 REM
1180 REM  RANDOM COLORI E SUONO
1190 REM
1200 FOR  K = 1  TO  30
1210 X = FNM(A) : Y = FNM(A)
1220 X1 = C(X)
1230 C(X) = C(Y) : C(Y) = X1
1240 X = FNM(A): Y = FNM(A)
1250 X1 = D(X)
1260 D(X)= D(Y) : D(Y) = X1
1270 NEXT
1280 R$ = "[HOME]" : REM  CLR-HOME
1290 FOR  K = 1  TO  5 : R$ = R$ +
      "[DOWN]" : NEXT
1300 FOR  K = 1  TO  32 : R$ = R$
      + "[RIGHT]" : NEXT
```

Le immagini di questo fascicolo

Charles Babbage è considerato il padre dei computer per una macchina mai realizzata. Il progetto della macchina analitica (1832(conteneva però uno schema di elaborazione dei dati identico e quello dei modemi computer.

Charles Babbage, 1791-1871

```
1310 R1$ = R$ + "[2 DOWN][BIANCO]"
     : F$ = R$ + "[AZZUR]"
1320 R2$ = R1$+"[3 DOWN][VERDE2][2
     RIGHT]"
1330 R3$ = R1$ + "[2 DOWN][VERDE2]
     "
1340 R4$ = R3$ + "[3 DOWN][VERDE]"
1350 R5$ = R4$ + "[DOWN][2 RIGHT]"
1360 TI$ = "000000" : T$ = "000000
     "
1370 REM
1380 REM  INIZIO ROUTINE PRINCIPAL
     E
1390 REM
1400 GOSUB  1770 : FL = 0 : R1 = 0
1410 X = 32 : Y = 58 : X1 = X : Y1
     = Y
1415 LV = -1 : GOSUB  2060
1420 P=PEEK(56321) : POKE  2040,OM
1430 IF P=254 THEN 1510
1440 IF P=251 THEN 1570
1450 IF P=247 THEN 1630
1460 IF P=253 THEN 1690
1470 REM   IFFL=0THENGOSUB2060
1480 IF TI$>T$ THEN 2130
1490 PRINTR1$TI$
1500 GOTO 1420
1510 REM  IN ALTO
1511 IF OM=13 THEN OM=14:GOTO 1520
1512 OM=13
1520 E=FNC(A):POKE W,FNM(A)*10:POK
     E W,0
1530 GOSUB 20000
1540 IF E>99 OR E<35 THEN 1420
1550 Y=Y-8
1560 POKE V,X:POKE V+1,Y:GOTO 1420
1570 REM  A SINISTRA
1571 IF OM=13 THEN OM=14:GOTO 1580
1572 OM=13
1580 E=FNE(A):POKE W,FNM(A)*10:POK
     E W,0
1590 GOSUB 20000
1600 IF E>99 OR E<35 THEN 1420
1610 X=X-8
1620 POKE V,X:POKE V+1,Y:GOTO 1420
1630 REM  A DESTRA
1631 IF OM=13 THEN OM=14:GOTO 1640
1632 OM=13
1640 E=FNF(A):POKE W,FNM(A)*10:POK
     E W,0
1650 GOSUB 20000
1660 IF E>99 OR E<35 THEN 1420
1670 X=X+8
1680 POKE V,X:POKE V+1,Y:GOTO 1420
1690 REM  IN BASSO
1691 IF OM=13 THEN OM=14:GOTO 1700
1692 OM=13
1700 E=FNH(A):POKE W,FNM(A)*10:POK
     E W,0
1710 GOSUB 20000
1720 IF E>99 OR E<35 THEN 1420
1730 Y=Y+8
1740 POKE V,X:POKE V+1,Y:GOTO 1420
1750 GOTO 1420
1760 REM  GIOVANNI BELLU' SOFTWARE
      1984
1770 PRINT"[CLEAR][VERDE2]";
1780 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
     ▒▒▒▒▒▒▒▒▒▒▒"
1790 PRINT"▒
         ▒"
1800 PRINT"▒ ♠--------------------
     ------ ▒"
1810 PRINT"▒ -        -    -    -    -
        - ▒"
1820 PRINT"▒ - ▒▒▒▒▒▒ - ▒ - ▒ - ▒ -
     ▒▒▒▒ - ▒"
1830 PRINT"▒ - ▒    ▒ - ▒ - ▒ - ▒ -
     ▒ ▒ * ▒"
1840 PRINT"▒ - ▒ $ ▒ - ▒ -     - ▒ -
        ▒ - ▒"
1850 PRINT"▒ - ▒ - ▒ - ▒ --●-- ▒ -
     -# ▒ - ▒"
1860 PRINT"▒ - ▒ - ▒ - ▒ -     - ▒ -
        ▒ - ▒"
```

```
1870 PRINT"▒ - ▒ - ▒ - ▒ - ▒ - ▒ -
     ▒ ▒ - ▒"
1880 PRINT"▒ -    - ▒ - ▒ - ▒ - ▒ -
     ▒▒▒ - ▒"
1890 PRINT"▒ ----- ▒ O ▒ - ▒ -    -
     - ▒"
1900 PRINT"▒ -       ▒ - ▒ - ▒ -----
     -----♦ ▒"
1910 PRINT"▒ - ▒▒▒▒▒ - ▒ - ▒ -    -
     - ▒"
1920 PRINT"▒ -          -    -    - ▒ -
     ▒▒▒ - ▒"
1930 PRINT"▒ ---------------- ▒ -
     ▒▒▒ - ▒"
1940 PRINT"▒      -         -     ▒ -
     - ▒"
1950 PRINT"▒▒▒▒▒ - ▒▒▒▒▒▒ + ▒▒▒ -
     ------ ▒"
1960 PRINT"▒      -         -       -
     - ▒"
1970 PRINT"▒ --------------------
     ▒▒▒ - ▒"
1980 PRINT"▒ -                       -
     ▒▒▒ - ▒"
1990 PRINT"▒ - ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒ -
     - ▒"
2000 PRINT"▒ ♥ ▒    GIOVANNI    ▒ -
     -----♣ ▒"
2010 PRINT"▒    ▒  BELLU' 1985   ▒
     ▒"
2020 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
     ▒▒▒▒▒▒▒[HOME]";
2030 PRINTR3$"LIVELLO"
2040 PRINTR4$" PUNTI "
2050 RETURN
2060 FL = 1 : TI$ = "000000" : LV
     = LV + 1
2070 I  = INT(LV/5) + 1 : IF  I >
     6  THEN  2210
2080 T$ = B$(I)
2090 PRINT R$ T$ : PRINT R2$ I : P
     RINT R5$ LV
2110 RETURN
2120 POKE W,0:POKE 53280,0:FL=0:TI
     $="000000":GOTO 1420
2130 REM
2140 REM   RIFLESSI ... LENTI!
2150 REM
2160 PRINTR1$"[RVS]"T$
2170 FOR J=1 TO 9:FOR K=1 TO 255 S
     TEP 5:POKE W,K:NEXT
2180 FOR K=255 TO 1 STEP -5:POKE W
     ,K:NEXT
2190 NEXT
2200 POKE W,0:PRINT"[CLEAR]":POKE
     V+21,0:GOTO 2230
2210 POKE V+21,0:PRINT"[CLEAR][RVS
     ] HAI DEI RIFLESSI VERAMENTE
     "
2220 PRINT"[RVS] ECCEZIONALI ... !
     "
2230 PRINT"ANCORA ?"
2240 GET A$:IF A$<>"S" AND A$<>"N"
      THEN 2240
2250 IF A$="S" THEN  RUN
2260 IF A$="N" THEN  END
2270 REM  GIOVANNI BELLU' SOFTWARE
      1985
10000 REM
10001 REM  RANDOM SEQUENZA
10002 REM
10010 FOR  K = 1  TO  10
10011 S(K,3)=FNM(A)
10015 FOR  J = 1  TO  2
10020 SK  =  INT(RND(1)*10+1)
10030 IF  SK < 1  OR  SK > 10  THEN
        10020
10040 S(K,J) = A(SK)
10041 NEXT J,K
10999 RETURN
20000 FOR J=1 TO 10:IF E=A(J) THEN
      J=10:NEXT:GOTO 20011
20001 REM  1778
20010 NEXT:RETURN
20011 IF R1=0 THEN POKE 1778,E:POKE
       1779,32:GOTO 20020
20012 POKE 1778,R1:POKE 1779,E:GOTO
       20020
20020 IF R1=0 THEN R1=E:GOSUB 21000
      :RETURN
20030 FOR K=1 TO 5
20040 IF S(K,1)=R1 AND S(K,2)=E  TH
      EN J=K:K=10:NEXTK:GOTO 20060
20050 NEXT:R1=0:POKE 53280,0
20051 POKE 1778,32:POKE 1779,32:GOT
      O 20100
20060 REM
20061 S(J,3)=S(J,3)-1:POKE 1778,32:
      POKE 1779,32
20062 IF S(J,3)=0 THEN S(J,1)=A(FNM
```

```
      (A)):S(J,2)=A(FNM(A))
20063 IF S(J,3)=0 THEN S(J,3)=FNM(A
      )
20065 R1=0
20066 FOR K=1 TO 20:POKE W,FNM(A)*5
      :NEXT:POKE W,0:GOSUB 2060
20070 TI$="000000":RETURN
20100 FOR K=200 TO 0 STEP -5:POKE W
      ,K:NEXT
20110 T$=B$(RND(1)*6+1)
20120 PRINTR$T$:RETURN
21000 FOR K=1 TO 5
21010 IF S(K,1)=E OR S(K,2)=E THEN
      POKE 53280,1:K=5:NEXT:RETURN
21020 NEXT:POKE 53280,0:RETURN
50000 PRINT"[CLEAR]":NL=60220
50010 PR=14
50020 FOR K=0 TO 63 STEP 6:PRINTNL"
      DATA";
50030 FOR X=0 TO 5
50040 A=PEEK(PR*64+K+X)
50050 PRINTA",";
50060 NEXTX:NL=NL+10
50070 PRINT"[2 LEFT]  ":NEXT
60110 DATA  0 , 0 , 0 , 0 , 0 , 0
60120 DATA  0 , 170 , 0 , 0 , 170 ,
       160
60130 DATA  0 , 165 , 0 , 0 , 89 ,
      64
60140 DATA  0 , 90 , 0 , 0 , 42 , 0
60150 DATA  1 , 127 , 0 , 3 , 255 ,
      20
60160 DATA  1 , 95 , 208 , 0 , 255
      , 0
60170 DATA  3 , 170 , 176 , 3 , 242
      , 240
60180 DATA  0 , 240 , 192 , 0 , 0 ,
      0
60190 DATA  0 , 0 , 0 , 0 , 0 , 0
60200 DATA  0 , 0 , 0 , 0 , 0 , 0
60210 DATA  0 , 0 , 0 , 0 , 0 , 0
60220 DATA  0 , 0 , 0 , 0 , 170 , 0
60230 DATA  0 , 170 , 160 , 0 , 165
      , 0
60240 DATA  0 , 89 , 64 , 0 , 90 ,
      0
60250 DATA  0 , 42 , 0 , 1 , 127 ,
      0
60260 DATA  3 , 255 , 80 , 1 , 95 ,
      64
60270 DATA  3 , 255 , 0 , 0 , 170 ,
      0
60280 DATA  0 , 170 , 0 , 0 , 255 ,
      192
60290 DATA  0 , 255 , 192 , 0 , 0 ,
      0
60300 DATA  0 , 0 , 0 , 0 , 0 , 0
60310 DATA  0 , 0 , 0 , 0 , 0 , 0
60320 DATA  0 , 0 , 0 , 0 , 0 , 0
```

Le immagini di questo fascicolo

Tempi moderni, il capolavoro di Charlie Chaplin. La storia dei computer è anche quella dell'uomo, e di un modo diverso di lavorare. Ancora adesso, dalle macchine ci si aspetta la liberazione dal lavoro manuale.

PUNTATORI

DATA MAKER

Il programma consente di trasformare una zona qualsiasi di memoria in linee basic DATA

Di solito, le brevi routine scritte in assembler, devono essere trasformate nei loro rispettivi codici numerici decimali. Per poter essere caricate in memoria dal programma Basic principale. Questo comporta un non indifferente quanto noioso lavoro, in quanto una piccola disattenzione può in seguito portare irreparabili danni al normale funzionamento del programma.

Al RUN il programma chiederà l'indirizzo di partenza e quello di fine che vogliamo trasformare in linee data dopodiché il numero di linea da cui devono partire le istruzioni DATA (che deve essere maggiore di 30).

Dopo un breve periodo di tempo, che dipende da quante locazioni di memoria devono essere trasformate, apparirà automaticamente il LIST delle linee DATA che sono state create.

Ma il bello viene ora: se infatti andiamo a fare il LIST, vedremo che il programma 'DATA MAKER' sarà misteriosamente scomparso ed avremo solamente le linee DATA che interessano e che possono essere quindi salvate su nastro o su disco per poi agganciarle al programma cui erano destinate (articolo "APPEND" pag. 37 n. 15 C.C.C.)

A questo punto vi sarete chiesti che fine ha fatto il... programma scomparso. È presto detto! Il trucco consiste nel manipolare i puntatori di inizio Basic, contenuti nelle locazioni 43-44 ($002b-$002c), in modo tale da spostarli all'inizio delle linee DATA che si saranno formate alla fine della fase di trasformazione.

Come conoscere tale indirizzo?

Anche in questo caso chiediamo aiuto ad un altro puntatore di pagina zero e precisamente al 65-66 ($0041-$0042). Tale puntatore contiene infatti l'indirizzo del DATA prossimo alla lettura.

Poiché detto puntatore viene inizializzato dopo la prima READ eseguita, ecco che la linea 200 pensa a fare questo decrementando poi tale indirizzo di 9 per ottenere l'indirizzo effettivo della prima linea DATA.

È chiaro che spostando il puntatore di inizio Basic all'indirizzo trovato in questo modo, non faremo altro che nascondere il programma precedente rendendo visibili solamente le linee DATA che interessano.

Detto questo chi è capace di far 'risorgere' il programma DATA MAKER?

E visto che siamo in tema di indovinelli, peraltro molto semplici, chi sa dire a che cosa serve la variabile "O" definita nella linea 180?

La subroutine in linea 290 consente, tramite il passaggio dei parametri X, LB%, HB% di trasformare un indirizzo nel suo Byte Alto e Basso da poter "Pokare" nei

puntatori.

I puntatori utilizzati dal DATA MAKER sono 6 locazioni di pagina zero:

251-252 = contengono il valore della locazione di partenza (linea 150)

252-253 = contengono il valore della locazione finale (Linea 155)

249-250 = contengono il numero della linea data da creare (Linea 140)

(Per default il numero delle linee DATA viene incrementato di uno, ma basta cambiare l'incremento della variabile LN nella linea 18 del programma per avere uno 'step' maggiore).

L'introduzione delle linee DATA è fatta mediante l'ormai sfruttatissima utilizzazione del buffer di tastiera con la forzatura del RETURN (linee da 27 a 29).

A chi fosse interessato a tale argomento consiglio di rileggere l'articolo. 'Il buffer della tastiera' pubblicato sul numero 8, naturalmente sul Vostro... Commodore Computer Club.

Giovanni Beani

```
100 REM * DATA MAKER
110 INPUT"▯INDIRIZZO DI PARTENZA";SA:
    IFSA<8THEN110
120 INPUT"INDIRIZZO DI FINE";EA
125 IFEA<=SA OR EA >65535 THENPRINT"
    ▯▯";:GOTO120
130 INPUT"INIZIO NUMERAZIONE BASIC";LN
132 IF LN<31 OR LN>63000 THEN"▯▯";:GOTO130
140 X=LN-1:GOSUB290:POKE249,LB%:POKE250,HB%
150 X=SA-7:GOSUB290:POKE251,LB%:POKE252,HB%
155 X=EA:GOSUB290:POKE253,LB%:POKE254,HB%
160 PRINT"▯"SPC(220)SPC(231)"
    ▯ATTENDERE, PREGO"
170 LN=PEEK(249)+PEEK(250)*256:LN=LN+10:X=LN
175 GOSUB290:POKE249,LB%:POKE250,HB%
180 O=1000:X=PEEK(251)+PEEK(252)*256+7
185 GOSUB290:POKE251,LB%:POKE252,HB%
190 IF X<PEEK(253)+PEEK(254)*256 THEN 220
200 PRINT"▯";:READ SA:SA=PEEK(65)
    +PEEK(66)*256-9
210 X=SA:GOSUB290:POKE43,LB%:POKE44,HB%:LIST
220 FORJ=0TO6
230 D=PEEK(X+J):D1$=RIGHT$(STR$(O+D),3)
240 D2$=D2$+D1$+","
250 NEXT
260 PRINT"▯▯▯"LN"DATA ";LEFT$(D2$,LEN(D2$)-1)
270 PRINT"GOTO170▯"
280 POKE198,2:POKE631,13:POKE632,13:END
290 HB%=X/256:LB%=X-HB%*256:RETURN
READY.
```

LINGUAGGIO MACCHINA

QUINTA PARTE: L'INDIRIZZAMENTO

Ricapitoliamo i concetti di base esaminati nelle quattro puntate precedenti. Chiunque abbia provato, almeno una volta, a creare loop (iterazioni) di una certa lunghezza in Basic per eseguire grafici di funzione o per trasferire dati all'interno della memoria stessa, si è irrimediabilmente scontrato con una lentezza esasperante, tale da far desistere il più volenteroso degli operatori. Il motivo di tale velocità limitata è l'interprete Basic, che richiede un numero notevole di istruzioni in linguaggio macchina per la più semplice delle procedure.

Ma perchè allora non lo aggiriamo? Nasce dunque la necessarietà di capire come funziona il microprocessore che utilizza il Commodore 64, il 6510 (oppure il 6502). Come già sappiamo, questo circuito integrato, definito più genericamente CPU (dall'inglese Central Process Unit), lavora ad otto bit, vale a dire che la lunghezza di ogni dato o istruzione non può superare le otto cifre in binario, equivalenti al fatidico 256 in notazione decimale.

Fig. 1. Indirizzamento indiretto.

Fig. 2. Indirizzamento indicizzato indiretto.

All'interno del 6502 sono contenuti alcuni registri, tutti ad 8 bit, utilizzabili per la memorizzazione temporanea dei dati transistori (A,X,Y), o per la verifica dello stato del processore (P). Unica eccezione, se così può essere definita, è costituita dal contatore di programma (PC) che è un registro a 16 bit, il cui scopo è quello di tenere il conto del passo di programma che viene eseguito e di valutare il successivo.

In realtà si tratta di due registri ben distinti (PCL, PCH) che, abbinati, sono in grado di indirizzare fino ad un massimo di 65535 locazioni di memoria (16 bit)

vedi figura 2.

Questa tecnica di indirizzamento composta non ci è nuova, ma è sempre utile ricordare che i numeri compresi fra 256 e 65535 vengono immagazzinati in due registri successivi chiamati per convenzione LB (Low Byte) ed HB (High Byte) nell'ordine LB,HB. Il loro valore globale è dato dalla formula:

ADDR = LB + 256*HB (Fig. 3).

Analizziamo le istruzioni che già conosciamo. Nelle precedenti puntate (pubblicate sui numeri 6, 8, 13 e 17 di Commodore Computer Club) abbiamo analizzato le istruzioni più importanti ed alcuni indirizzamenti tipici del 6502 (& 6510). Prima di riesaminarle molto velocemente, è bene ricordare che le istruzioni sono in tutto 56, mentre gli indirizzamenti disponibili sono 13; comunque nessuna istruzione prevede l'utilizzazione di tutti questi, cosicchè alla resa dei conti abbiamo un set di 151 istruzioni con tutte le variabili possibili.

A seconda dell'operazione che svolgono, le istruzioni si dividono in cinque settori:

● **Trasferimento dati.** Caricano i registri con dati nella memoria (LDA LDX LDY). Caricano la memoria con il contenuto dei registri (STA STX STY). Interscambiano i registri tra di loro (TAX TAY TYA TXA).

● **Elaborazione dati.** Eseguono le funzioni aritmetiche e logiche, incrementano e decrementano registri e memoria (INC DEC INX DEY).

● **Confronti test e diramazioni.** Eseguono confronti fra registri e memoria (CPX CPY CMP). Eseguono diramazioni in seguito a confronti con il registro di stato P (BMI BPL BCC BCS BEQ BNE BVS BVC).

● **Controllo.** Modificano il registro di stato P (CLC SEC CLD SED CLV).

● **Flusso di programma.** Eseguono salti con o senza ritorno (JMP JSR RTS).

Metodi di indirizzamento

Analizzate le istruzioni di base, ricapitoliamo i metodi di indirizzamento, quelli cioè che determinano il campo nel quale agisce una determinata istruzione:

● **Implicito** (1 byte) riguarda per lo più la gestione interna dei registri (TAX INX), ma si può occupare anche della gestione del flag di stato P (CLC); non richiede in nessun caso che vengano specificati parametri.

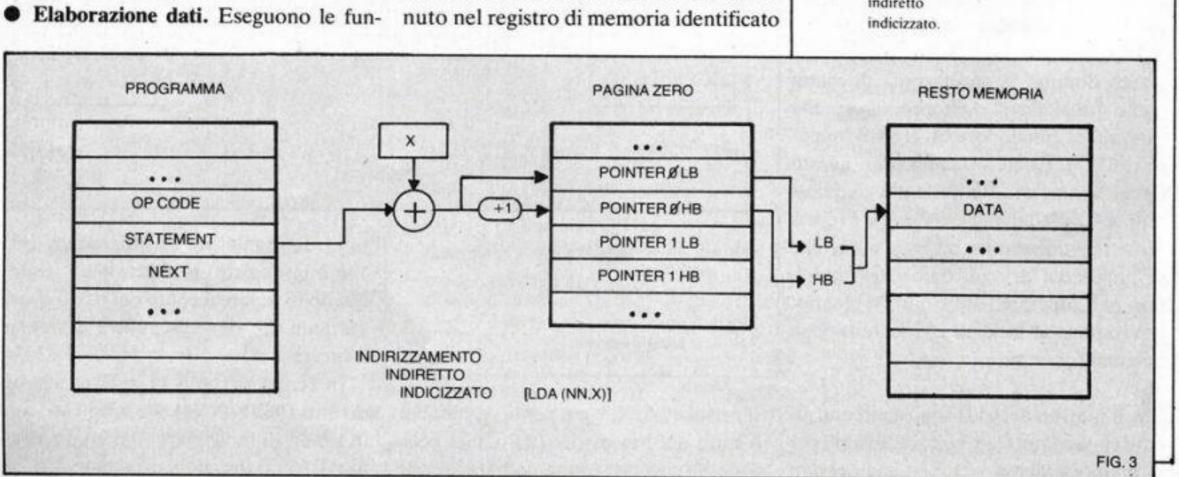
● **Immediato** (2 byte) l'operazione indicata dal codice operativo viene eseguita sul numero che lo segue immediatamente (esempio: LDA · $00). L'operando, naturalmente non può essere maggiore di 255.

● **Assoluto** (3 byte) L'operando che deve essere trattato dall'istruzione è contenuto nel registro di memoria identificato dai 2 byte che seguono il codice operativo stesso (esempio: LDA $A100); i due byte sono nella solita disposizione LB, HB.

● **Pagina zero** (2 byte) variante dell'indirizzamento assoluto, pone automaticamente HB uguale a 0, limitando così il campo dell'operazione alle prime 256 locazioni di memoria che peraltro contengono dati molto importanti per il funzionamento del calcolatore (LDA $F0). Tale registrazione aumenta ovviamente la velocità di esecuzione.

● **Relativo** (2 byte) la parola che segue l'istruzione indica uno spostamento, in avanti di 128 parole ed all'indietro di 127, secondo uno standard piuttosto complesso (esempio: BEQ $t12). In ogni caso provvede ai debiti calcoli l'eventuale assemblatore.

● **Indicizzato** (2 byte se in pagina zero. 3 byte se assoluto). L'indirizzo sul quale deve agire l'istruzione viene ottenuto dalla somma del dato (o dei dati se assoluto) posto dopo il codice operativo, con il registro interno (solo X e Y) specificato nell'istruzione stessa (LDA $A001,Y). Se abbiamo per esempio l'istruzione LDA $C000,Y (in cui Y=$05), la locazione di memoria che verrà posta in A sarà $(C000 + 05) =$C005.

Fig. 3. Indirizzamento indiretto indicizzato.

L'indirizzamento indiretto

Il termine "indiretto", tecnicamente deriva dal fatto che invece di prevalere l'indirizzo che segue immediatamente l'istruzione, il successivo valore nella sequenza di programma è un puntatore all'indirizzo interessato. Cerco di chiarire meglio il concetto.

Nelle pagine 0, 1, 2, 3, cioè nelle locazioni di memoria che vanno da 0 a 1023, (appena "sotto" il video del C-64) sono contenuti un gran numero di puntatori, che "puntano" per l'appunto a routines nella ROM fondamentali per il funzionamento del Commodore 64. E' anche notó che questo fatto aumenta notevolmente le doti di flessibilità del calcolatore, perchè permette all'utente di modificare a proprio piacimento tutti o quasi i procedimenti normalmente eseguiti automaticamente.

A questo punto è lecito chiedersi come faccia il 6502 a prelevare l'indirizzo indicato nei puntatori. In effetti, con le conoscenze che abbiamo a disposizione, questo non ci è permesso, a meno di complicazioni assurde che resentano il limite della correttezza e della linearità di programmazione.

Ma gli "architetti" del 6502 hanno pensato anche a questo, creando un'unica istruzione che sfrutti l'indirizzamento indiretto non indicizzato (più oltre analizzeremo l'indicizzazione abbinata all'indirezione), ma si rivela di importanza fondamentale soprattutto nella gestione della ROM a cui accennavamo prima: si tratta di "JMP (XXXX)" (op code 6C XX XX).

Il funzionamento è abbastanza semplice da capire, ma occorre prestare sempre un certa attenzione ai calcoli da eseguire in fase di stesura del programma e di modifica dei puntatori. Infatti in seguito a questa istruzione il P.C (Contatore di Programma) viene modificato in modo tale che il Low Byte sia caricato con la parola contenuta nella locazione di memoria indicata dall'istruzione, e che l'High Byte sia caricato con il contenuto della locazione immediatamente seguente a quella indicata.

In altre parole data l'istruzione JMP($0306) e supponendo che ($0306) contenga ($1A) e che ($0307) contenga ($A7), il P.C. si posizionerà all'indirizzo $A714 (vettore di List). L'utilità pratica del fatto di avere in un'area RAM (e quindi modificabile) tutti i vettori relativi alle funzioni più importanti è basilare, perchè permette di alterare tali routines, implementandole o addirittura sostituendole con altre create all'utente.

L'indicizzazione abbinata all'indirezione

Accennavamo prima alla possibilità di abbinare l'indicizzazione alla tecnica dell'indirezione. Bene, fin d'ora è bene chiarire che esistono due soluzioni ben differenti fra di loro nonostante la buffa (ma non troppo) caratteristica di chiamarsi pressapoco allo stesso modo (indirezzamento indicizzato indiretto e indiretto indicizzato). La caratteristica pratica che li contraddistingue è che la prima di queste tecniche sfrutta solo ed esclusivamente l'indice Y, mentre la seconda sfrutta X.

Prima di osservare le ulteriori differenze, è indispensabile osservare che entrambe "lavorano" solo in pagina zero, apetto questo da tenere bene a mente. L'indirizzamento indicizzato indiretto (quello che sfrutta Y, per intenderci) si comporta nel modo seguente:

L'indirizzo effettivo dell'istruzione è dato dal contenuto della locazione in pagina 0 (L.B.), più il contenuto della locazione seguente moltiplicato per 256 (H.B.), più il registro Y (indice). In altre parole, i due bytes in pagina 0 (dei quali viene indicato solo il primo), sono dei puntatori ad una tabella di 256 elementi locata in qualsiasi area della memoria, che può essere scorsa per intero modificando solo l'indice Y (INY, DEY).

Un esempio:
LDA($F0,Y) dove $F0 contiene [$16] e

$F1 contiene [$C0], con Y=$03, carica l'accumulatore con il contenuto della locazione di memoria $[C016+03]=$C19.

Nel caso dell'indirizzamento indiretto indicizzato (con X come indice), le cose sono lievemente diverse. L'indice, infatti, viene aggiunto immediatamente all'indirizzo specificato dall'istruzione, per puntare ad un altro indirizzo indiretto (espresso in LB, HB) sempre in pagina 0. E' importante rilevare che la somma non tiene conto del riporto, quindi se il risultato supera il limite della pagina la locazione considerata è quella risultante diminuita di 256. Tale caratteristica permette di creare una tabella di puntatori allocati in pagina 0 che può essere analizzata modificando semplicemente l'indice X (INX, DEX).

Ad esempio:

LDA($F0,X) carica in A la parola il cui L.B. è dato dal contenuto di $[F0+X] ed l'H.B. dal contenuto di $[F1+X].

Ricapitolando, l'indice Y influisce sull'indirizzo effettivo, mentre l'indice X modifica direttamente i puntatori in pagina 0. Ecco spiegata la strana somiglianza dei due nomi. Dall'uso contemporaneo di entrambe le tecniche risultano vantaggi immediatamente apprezzabili che permettono finalmente di lavorare in linguaggio macchina con una certa linearità e disinvoltura.

Applicazioni

Per quanto riguarda le applicazioni, i programmi proposti si commentano da soli.

□ **Il programma 1** esegue esattamente la stessa operazione dell'ultimo programma presentato nella puntata precedente, e cioè trasferisce i messaggi di errore ed i tokens del Basic sul video, solo che sfrutta la tecnica dell'indirizzamento indicizzato indiretto (Y).

La prima parte costituisce la necessaria inizializzazione, che permette di ripetere più volte l'esecuzione senza incorrere in grossolani errori. Tutti i valori interessati ($A000, $0400, $D800) vengono sistemati in pagina 0.

Nel nostro caso vengono locati in un'area destinata al floating point, e che quindi va adoperata solo quando non si usa il FLP. In ogni caso a lato è riportata una tabella di locazioni libere o per lo meno utilizzabili in pagina 0. Il registro X indica il numero di blocchi (da 256 bytes l'uno) che devono essere trasferiti, mentre il registro Y contiene l'eventuale OFFSET, cioè il numero il bytes (inferiori a 256) che devono essere trattati oltre ai precedenti blocchi.

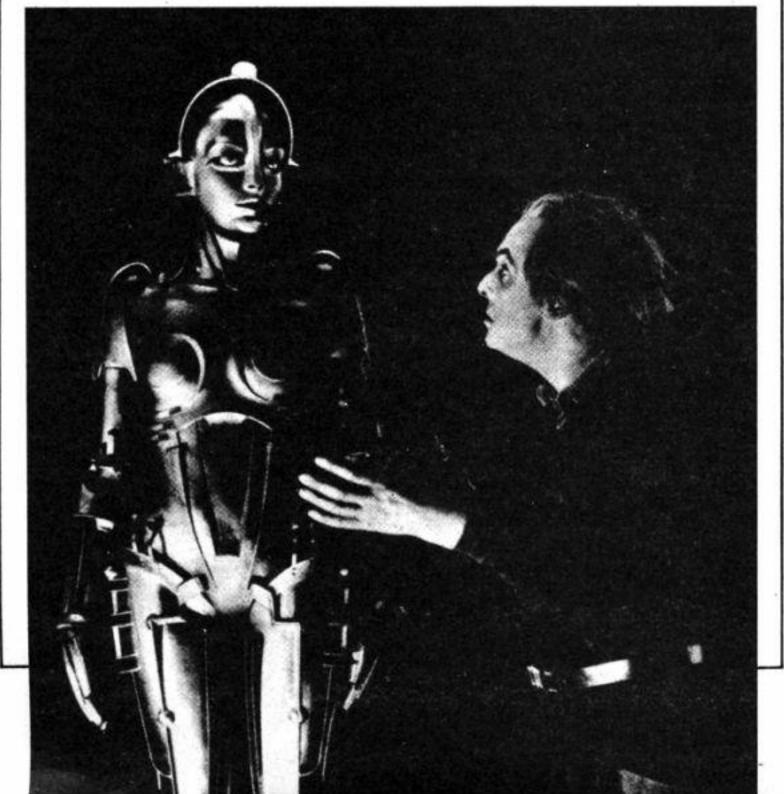
La subroutine si è resa necessaria per meglio identificare la parte centrale del programma, che del resto è già chiara per conto suo: analizzandola ci si rende effettivamente conto della funzione di X come contatore per 256 e della funzione di Y come indice.

□ **Il programma 2** non è altro che una variazione sul tema del programma precedente. La sua caratteristica è quella di sfruttare una subroutine in ROM che effettua il trasferimento dei dati ($A3BF). Tale procedura richiede come dati ingresso: la locazione d'inizio dei dati da trasferire (5F,60), la locazione finale dei dati da trasferire (5A,5B) e la locazione finale della destinazione dei dati (58,59).

I valori in questione vengono forniti dalla parte iniziale, quindi si salta direttamente in ROM per tornare al Basic alla fine dell'esecuzione.

□ **Il programma 3,** inserito per scopi puramente dimostrativi, è una semplice impostazione della pagina e della sua pulizia, ricalcando esattamente la procedura

Le immagini di questo fascicolo

Il regista americano Fritz Lang presenta per la prima volta nel suo film Metropolis del 1927 la prospettiva di un mondo dominato dalle macchine. Per la prima volta si discute a tutti i livelli sul ruolo e la funzione delle macchine.

Basic. Differenze? Provate a cronometrare quanto tempo ci mette, se ci riuscite!

□ **Il programma 4** è un'applicazione di entrambe le tecniche di indicizzazione. La prima parte serve per inizializzare i puntatori, trasferendo in pagina zero i 6 bytes che seguono immediatamente il programma stesso. Per fare ciò si sfrutta l'indicizzazione indiretta, anche se obbiettivamente se ne potrebbe fare a meno; ciononostante è importante rilevare la tecnica rilevata, cioè piazzare i puntatori in fondo al programma per poi riutilizzarli quando serve: in caso di indirizzi più numerosi, la tecnica è fondamentale.

Dopo l'inizializzazione vengono chiamate due sottoprocedure della ROM, la SCNKEY ($EA87) che serve per leggere la tastiera, e la GETIN ($F13E) che riporta in A il valore ASCII del tasto premuto. Per i controlli si sottae $30 (il numero 1 corrispondente al codice ASCII $31. A seconda sia stato premuto il tasto 1, 2 o 3, un messaggio diverso viene stampato sullo schermo; premendo il tasto 0 il programma termina. Per scrivere sullo schermo si sfrutta un'altra procedura della ROM, la CHROUT ($F1CA) che stampa sullo shermo il carattere il cui codice ASCII è contenuto in A. Quest'ultima parte sfrutta l'indirizzamento indiretto indicizzato (con X) per puntare al messaggio desiderato, che si può trovare ovunque nella memoria, e la cui lunghezza è indicata da Y.

Naturalmente questa serie di programmi vuole essere solo una base di partenza, una proposta che viene offerta all'utente per permettergli di escogitare nuove applicazioni e soluzioni sfruttando appieno la propria fantasia. Quindi, in bocca al lupo e...buon divertimento!

Simone Bettola

Locazioni esadecimali di pagina zero utilizzabili dall'utente

19-21 stack tempor. per stringa
22-25 area di utilizzo puntatori
26-2A risultato FLP di molt. e div.
4E-52 FLP temporaneo
57-5B FLP temporaneo
5C-60 FLP temporaneo
61-68 FLP accumulatore 1
69-70 FLP accumulatore 2
FB-FE spazio libero per l'utente

LINGUAGGIO MACCHINA (1)

```
10 PRINT"[CLEAR][DOWN]|RASFERISCE
   SULLO SCHERMO DEL "
20 PRINT"COMMODORE 64 LE ISTRUZION
   I BASIC ED"
30 PRINT"I MESSAGGI DI CONTROLLO "
40 PRINT"SFRUTTANDO L'INDIRIZZAMEN
   TO INDICIZZATO"
50 PRINT"INDIRETTO (CON Y)[4 DOWN]
   "
60 PRINTCHR$(14)
100 READ A$:IF VAL(A$)<0 THEN PRINT
    "[RVS]SYS49152":END
110 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGH
    T$(A$,1))
120 IF X1>57 THEN X1=X1-55:GOTO 140
130 X1=X1-48
140 IF X2>57 THEN X2=X2-55:GOTO 160
150 X2=X2-48
160 POKE 49152+AD,X1*16+X2:AD=AD+1:
    GOTO 100
200 DATA A9,04,85,62,A9,A0,85,64
210 DATA A9,D8,85,66,A9,00,85,61
220 DATA 85,63,85,65,A2,03,A0,00
230 DATA 20,23,C0,A2,01,A0,E8,20
240 DATA 23,C0,60,A9,00,91,65,B1
250 DATA 63,91,61,88,D0,F5,E6,62
260 DATA E6,64,E6,66,CA,D0,EC,60
270 DATA -1
```

```
., C000  A9 04     LDA #$04
., C002  85 62     STA $62
., C004  A9 A0     LDA #$A0
., C006  85 64     STA $64
., C008  A9 D8     LDA #$D8
., C00A  85 66     STA $66
., C00C  A9 00     LDA #$00
., C00E  85 61     STA $61
., C010  85 63     STA $63
., C012  85 65     STA $65
., C014  A2 03     LDX #$03
., C016  A0 00     LDY #$00
., C018  20 23 C0  JSR $C023
., C01B  A2 01     LDX #$01
., C01D  A0 E8     LDY #$E8
., C01F  20 23 C0  JSR $C023
., C022  60        RTS
., C023  A9 00     LDA #$00
., C025  91 65     STA ($65),Y
., C027  B1 63     LDA ($63),Y
., C029  91 61     STA ($61),Y
., C02B  88        DEY
., C02C  D0 F5     BNE $C023
., C02E  E6 62     INC $62
., C030  E6 64     INC $64
., C032  E6 66     INC $66
., C034  CA        DEX
., C035  D0 EC     BNE $C023
., C037  60        RTS
```

LINGUAGGIO MACCHINA (2)

```
 10 PRINT"[CLEAR][DOWN]TRASFERISCE
    SULLO SCHERMO DEL "
 20 PRINT"COMMODORE 64 LE ISTRUZION
    I BASIC ED"
 30 PRINT"I MESSAGGI DI CONTROLLO "
 40 PRINT"SFRUTTANDO LE ROUTINES DE
    LLA ⌐⌐\"
 50 PRINTCHR$(14)
100 READ A$:IF VAL(A$)<0 THEN PRINT
    "[RVS]SYS49152":END
110 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGH
    T$(A$,1))
120 IF X1>57 THEN X1=X1-55:GOTO 140
130 X1=X1-48
140 IF X2>57 THEN X2=X2-55:GOTO 160
150 X2=X2-48
160 POKE 49152+AD,X1*16+X2:AD=AD+1:
    GOTO 100
200 DATA A9,00,85,5F,A9,A0,85,60
210 DATA A9,E8,85,5A,A9,A3,85,5B
220 DATA A9,E8,85,58,A9,07,85,59
230 DATA 20,BF,A3,60,-1
```

```
., C000  A9 00     LDA #$00
., C002  85 5F     STA $5F
., C004  A9 A0     LDA #$A0
., C006  85 60     STA $60
., C008  A9 E8     LDA #$E8
., C00A  85 5A     STA $5A
., C00C  A9 A3     LDA #$A3
., C00E  85 5B     STA $5B
., C010  A9 E8     LDA #$E8
., C012  85 58     STA $58
., C014  A9 07     LDA #$07
., C016  85 59     STA $59
., C018  20 BF A3  JSR $A3BF
., C01B  00        BRK
```

LINGUAGGIO MACCHINA (3)

```
 10 PRINT"[CLEAR][DOWN]IMPOSTA E PU
    LISCE LA PAGINA GRAFICA"
 20 PRINT"SFRUTTANDO L'INDIRIZZAMEN
    TO INDICIZZATO"
 30 PRINT"INDIRETTO (IN |), QUINDI
    DISEGNA UNA"
 40 PRINT"SINUSOIDE[4 DOWN]"
 50 PRINTCHR$(14)
100 READ A$:IF VAL(A$)<0 THEN PRINT
    "[RVS]SYS49152:RUN170":END
110 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGH
    T$(A$,1))
120 IF X1>57 THEN X1=X1-55:GOTO 140
130 X1=X1-48
140 IF X2>57 THEN X2=X2-55:GOTO 160
150 X2=X2-48
160 POKE 49152+AD,X1*16+X2:AD=AD+1:
    GOTO 100
170 FOR X=0 TO 319
180 V=INT(100+90*SIN(X/51))
190 CH=INT(X/8):RO=INT(V/8)
200 LN=V AND 7:BI=7-(X AND 7)
```

```
., C000  AD 18 D0  LDA $D018
., C003  09 08     ORA #$08
., C005  8D 18 D0  STA $D018
., C008  AD 11 D0  LDA $D011
., C00B  09 20     ORA #$20
., C00D  8D 11 D0  STA $D011
., C010  A9 40     LDA #$40
., C012  85 FD     STA $FD
., C014  A9 20     LDA #$20
., C016  85 FC     STA $FC
., C018  A9 00     LDA #$00
., C01A  85 FB     STA $FB
., C01C  A8        TAY
., C01D  20 2E C0  JSR $C02E
., C020  A9 08     LDA #$08
., C022  85 FD     STA $FD
., C024  A9 04     LDA #$04
., C026  85 FC     STA $FC
., C028  A9 03     LDA #$03
., C02A  20 2E C0  JSR $C02E
```

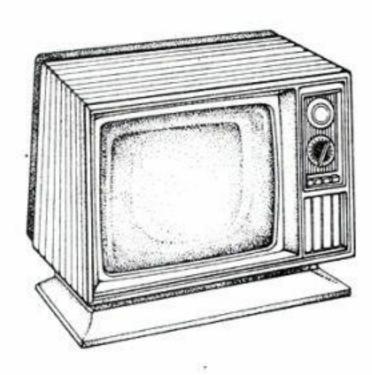
```
210 BY=8192+RO*320+8*CH+LN
220 POKE BY,PEEK(BY) OR (2↑BI)
230 NEXT:POKE 1024,16
240 GOTO 240
300 DATA AD,18,D0,09,08,8D,18,D0
310 DATA AD,11,D0,09,20,8D,11,D0
320 DATA A9,40,85,FD,A9,20,85,FC
330 DATA A9,00,85,FB,A8,20,2E,C0
340 DATA A9,08,85,FD,A9,04,85,FC
350 DATA A9,03,20,2E,C0,60,91,FB
360 DATA 88,D0,F8,E6,FC,A6,FC,E4
370 DATA FD,D0,F3,60,-1
```

```
., C02D  60       RTS
., C02E  91 FB    STA ($FB),Y
., C030  88       DEY
., C031  D0 FB    BNE $C02E
., C033  E6 FC    INC $FC
., C035  A6 FC    LDX $FC
., C037  E4 FD    CPX $FD
., C039  D0 F3    BNE $C02E
., C03B  60       RTS
```

LINGUAGGIO MACCHINA (4)

```
10 PRINT"[CLEAR][DOWN]♥CRIVE SULLO
   SCHERMO 3 MESSAGGI"
20 PRINT"DIFFERENTI A SECONDA SI P
   REMA 1, 2 O 3"
30 PRINT"E TERMINA CON 0 ."
40 PRINT"♥FRUTTA ENTRAMBE LE TECNI
   CHE DI"
50 PRINT"INDIREZIONE INDICIZZATA.[
   4 DOWN]"
60 PRINTCHR$(14)
100 READ A$:IF VAL(A$)<0 THEN PRINT
    "[RVS]SYS49152":END
110 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGH
    T$(A$,1))
120 IF X1>57 THEN X1=X1-55:GOTO 140
130 X1=X1-48
140 IF X2>57 THEN X2=X2-55:GOTO 160
150 X2=X2-48
160 POKE 49152+AD,X1*16+X2:AD=AD+1:
    GOTO 100
200 DATA A9,19,85,FB,A9,00,85,FC
210 DATA A9,41,85,FD,A9,C0,85,FE
220 DATA A0,05,B1,FD,91,FB,88,10
230 DATA F9,20,87,EA,20,3E,F1,38
240 DATA E9,30,F0,1C,30,F3,C9,04
250 DATA B0,EF,0A,AA,A0,0D,A1,17
260 DATA 20,CA,F1,F6,17,88,10,F6
270 DATA A9,8D,20,CA,F1,4C,00,C0
280 DATA 60,7D,E4,9B,E4,47,C0,42
290 DATA 45,54,54,4F,4C,41,20,53
300 DATA 49,4D,4F,4E,45,-1
```

```
., C000  A9 19     LDA #$19
., C002  85 FB     STA $FB
., C004  A9 00     LDA #$00
., C006  85 FC     STA $FC
., C008  A9 41     LDA #$41
., C00A  85 FD     STA $FD
., C00C  A9 C0     LDA #$C0
., C00E  85 FE     STA $FE
., C010  A0 05     LDY #$05
., C012  B1 FD     LDA ($FD),Y
., C014  91 FB     STA ($FB),Y
., C016  88        DEY
., C017  10 F9     BPL $C012
., C019  20 87 EA  JSR $EA87
., C01C  20 3E F1  JSR $F13E
., C01F  38        SEC
., C020  E9 30     SBC #$30
., C022  F0 1C     BEQ $C040
., C024  30 F3     BMI $C019
., C026  C9 04     CMP #$04
., C028  B0 EF     BCS $C019
., C02A  0A        ASL
., C02B  AA        TAX
., C02C  A0 0D     LDY #$0D
., C02E  A1 17     LDA ($17,X)
., C030  20 CA F1  JSR $F1CA
., C033  F6 17     INC $17,X
., C035  88        DEY
., C036  10 F6     BPL $C02E
., C038  A9 8D     LDA #$8D
., C03A  20 CA F1  JSR $F1CA
., C03D  4C 00 C0  JMP $C000
., C040  00        BRK
```

RASTER REGISTER ESPERIMENTI

Il raster register. Ovvero: come l'impossibile diventa possibile con un COMMODORE 64.

Il 64 consente una gestione del video molto completa e potente, normalmente non disponibile su macchine della sua fascia di prezzo. Sfruttare però a fondo le capacità di questo computer richiede uno studio approfondito della sua struttura.

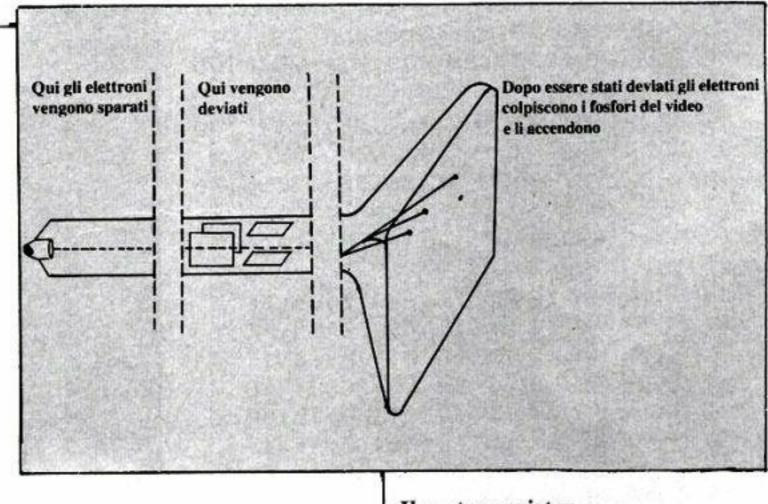
Obbligatorie quindi sono le procedure in linguaggio macchina, le lunghe letture di manuali e tabelle, le prove e riprove effettuate per vedere come la macchina reagisca in certe situazioni.
L'argomento trattato in questo articolo (ed in altri che seguiranno) richiede quindi una certa conoscenza del linguaggio assembly, anche se può essere inteso come un "propedeutico" invito per i meno esperti (purché sufficientemente volenterosi).

Probabilmente avrete tutti già letto l'articolo sugli interrupt pubblicato su C.C.C. n. 13: normalmente il computer interrompe l'esecuzione del programma principale circa una volta ogni sessantesimo di secondo, ovvero tutte le volte che si verifica un "underflow" del TIMER A che si trova in uno dei dispositivi di INPUT/OUTPUT del C.64 (il dispositivo in questione è il CIA 1, e per maggiori e migliori chiarimenti consultate la guida del Programmatore Commodore), consentendo la scansione della tastiera, mezzo tramite il quale voi siete soliti dialogare col computer.
Dovete però sapere che le sorgenti di interruzione sul 64 sono molte di più. Vale a dire, è possibile verificare situazioni "speciali" e trattarle come tali con appropriate procedure.
Nel nostro caso ci interessano solo le interruzioni del TIMER A e del circuito video.

Organizzazione del circuito video

Il circuito integrato VIC, che presiede le operazioni di uscita sullo schermo del 64, è molto versatile e potente, consentendo svariati tipi di visualizzazioni. Si alloca nella zona di memoria che va da 53248 (esa D000) fino a 53294 (esa D02E).
Suo tramite è possibile controllare fino ad 8 SPRITES, cambiare set di caratteri, passare in alta risoluzione eccetera.

Ebbene, grazie ad alcuni registri del VIC, è possibile far cose che normalmente sarebbero considerate impossibili (per fare un esempio, tenere visualizzati sullo schermo due set di caratteri contemporaneamente, oppure cambiare il colore di sfondo a metà schermo, oppure visualizzare più di otto SPRITES contemporaneamente).

FIG. 1

Il raster register

Questa "magia elettronica" è possibile grazie a 4 registri ed alla natura "sequen-

ziale" dell'immagine formata sullo schermo (figure 1, 2 e 3). Infatti l'immagine che vedete visualizzata su un video viene formata da un "cannone" di elettroni, il quale, cominciando dal margine in alto a sinistra e procedendo verso destra, traccia una linea dell'immagine (gli elettroni sparati dal cannone fanno illuminare i fosfori che sono distribuiti uniformemente sullo schermo).

Arrivato al margine destro, il cannone viene spento, e riposizionato a sinistra, ma una linea più in basso (la cosa a dire il vero non è proprio così, ma al fine di evitare inutili complicazioni evito una precisazione non necessaria per lo sviluppo di programmi utilizzanti le peculiarità del 64). Questo processo viene ripetuto, su un televisore PAL, 312 volte ogni cinquantesimo di secondo. Quando il cannone elettronico raggiunge il margine in basso a destra (là 312.ma linea), viene spento e riposizionato in alto a sinistra, ed il ciclo viene ripetuto.

FIG. 2 Le linee in neretto rappresentano le linee di immagine tracciati dal cannone elettronico. Come spiegato nell'articolo al termine di ogni linea il cannone viene spento e riposizionato al margine della linea seguente.

FIG. 3 N.B. Se il raster si trova tra la linea 0 e la linea 51 o fra la linea 251 e la linea 312, il cannone elettronico sta componendo l'immagine dell'area invisibile di schermo (bordo superiore o inferiore).

È possibile sapere in un qualsiasi momento su che linea si trovi il cannone consultando due registri del VIC:

il V+18 (53266, esa D012) che ritorna gli otto bit più bassi, ed il bit sette della locazione V+17 (dec 53265, esa D011) che ritorna il bit più significativo.

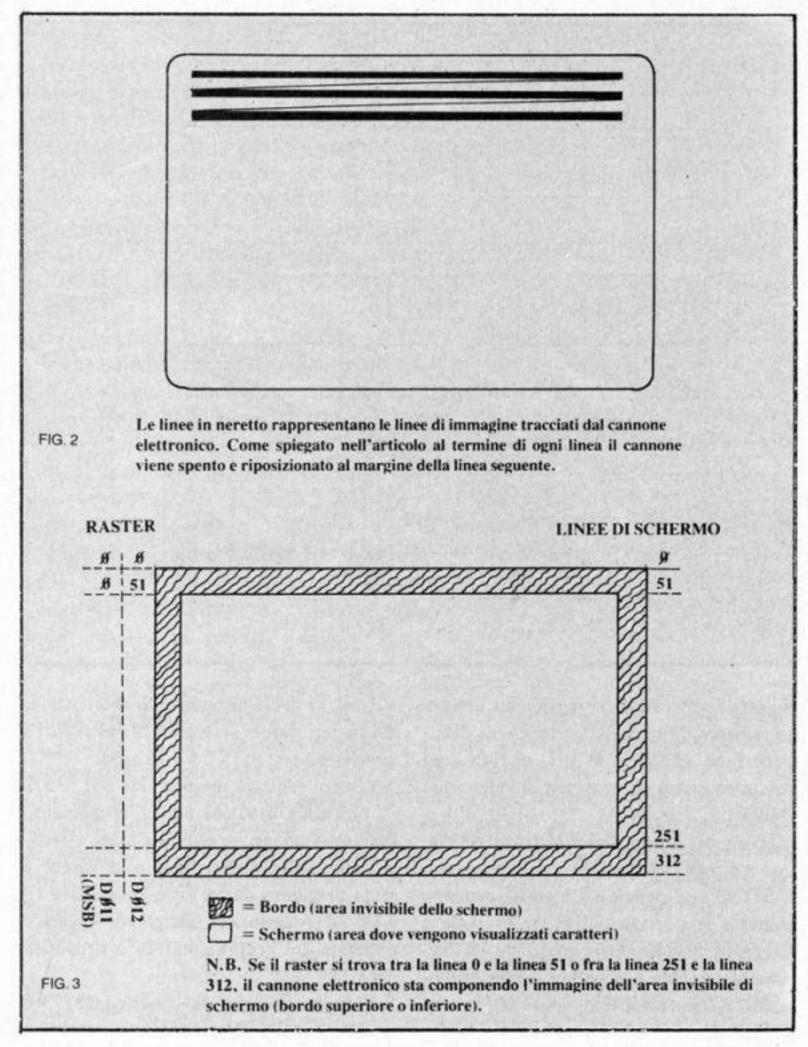
Infatti, il cannone si muove su 312 linee, e non è possibile rintracciare la sua posizione con una sola locazione di memoria (saprete che i numeri rappresentabili con un BYTE vanno da 0 a 255).

Quando il BIT sette della locazione 53265 (V+17) è attivato, vuol dire che il cannone ha superato la 255ma posizione e che gliene mancano ancora circa 56 per finire la composizione dell'immagine (figura 4).

Un pò di pratica

Provate un pò a digitare:

V=53248:POKE V+18,N:PRINT PEEK (V+18)

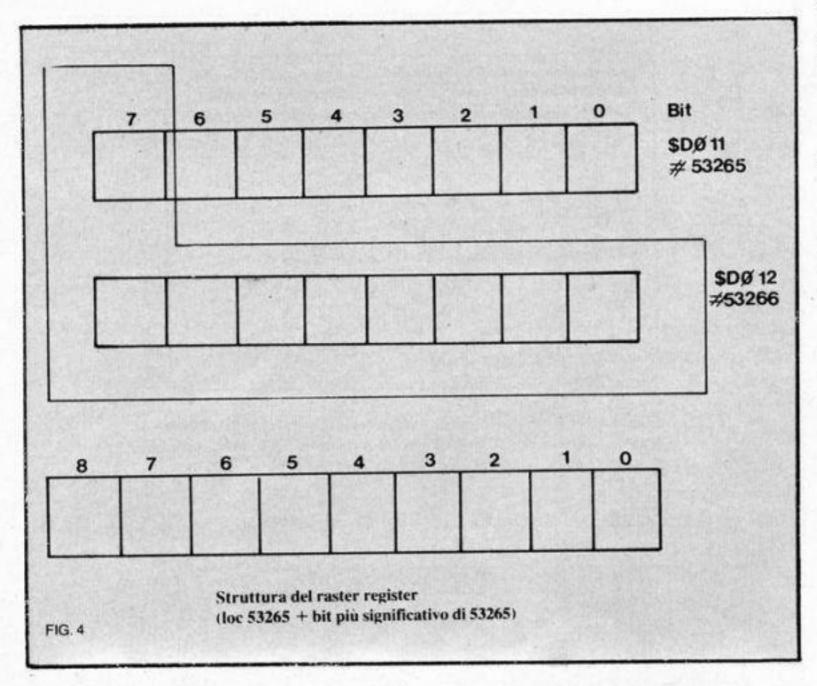
quasi sicuramente il valore che verrà stampato sarà diverso da quello che avevate scritto nella locazione (e questo per un qualsiasi N compreso tra 0 e 255).

Cosa sarà successo?: il computer ha ignorato quella istruzione e non ha eseguito la scrittura, oppure l'istruzione è stata eseguita ma non ha minimamente influenzato la locazione V+18?

Il valore che è stato introdotto nella locazione è stato effettivamente salvato, ma non è più visibile all'utente perché il registro in questione, in lettura, riporta sempre l'attuale posizione del cannone elettronico sullo schermo. Non appena si scrive un numero nel registro, il valore immesso viene conservato in un registro di comparazione.

Se sono state fatte altre procedure (che vedremo tra poco), non appena la posizione del cannone elettronico raggiunge quella precedentemente registrata, viene effettuata una interruzione mascherabile. Questo chiaramente permette all'utente di poter trattare lo schermo secondo le sue necessità, è di apportargli dei cambiamenti al posto giusto e nel momento voluto.

Maschere e interrupt

Prima si diceva che solo dopo ben precise procedure è possibile far generare una interruzione al circuito video al momento

Struttura del raster register
(loc 53265 + bit più significativo di 53265)

opportuno. Per far ciò è sufficiente settare un registro, la cosiddetta maschera di interruzione, che dice al VIC che da quel momento in poi gli interrupt di video sono abilitati.

L'organizzazione del registro (il 53274, vale a dire V+26 esa D01A) è la seguente:

BIT 0: se è uguale ad 1 allora vengono abilitate le interruzioni di raster (vale a dire: tutte le volte che la posizione del cannone elettronico è uguale a quella in precedenza registrata, viene generata una interruzione)

BIT 1: se è uguale ad 1, tutte le volte che uno SPRITE collide con lo sfondo, viene generata una interruzione

BIT 2: se è uguale ad 1, tutte le volte che uno SPRITE collide con uno SPRITE viene generata una interruzione

BIT 3: può essere sfruttato solo se è connessa una light pen.

BIT 4, 5, 6: non usati

BIT 7: bit di abilitazione. Di qualsiasi interruzione voi abbiate bisogno, ricordatevi SEMPRE ed in ogni caso di settare anche questo bit, che è appunto quello che abilita gli interrupt, di qualsiasi genere essi siano.

Così, se avete bisogno delle interruzioni del raster, dovete scrivere 129 nel registro abilitatore (es: POKE V+26,129).

L'altro registro in questione è il 53273 (V+25 esa D019) che indica, al momento dell'interruzione, se questa è stata effettivamente del circuito video o se invece è stata originata da un altro dispositivo (il CIA, ad esempio, e riconsiglio di leggere il manuale del programmatore a proposito del CIA).

L'organizzazione del registro 53273 è uguale a quella del precedentemente spiegato 53274, nel senso che:

BIT 0: se è a 1, vuol dire che il cannone elettronico ha raggiunto la posizione registrata precedentemente.

BIT 1: se è a 1, vuol dire che si è verificata una collisione tra SPRITE e sfondo

BIT 2: se è a 1, vuol dire che si è verificata una collisione SPRITE SPRITE

BIT 3: sfruttabile sempre e solo se una light pen è connessa%

BIT 4, 5, 6: non usati

BIT 7: a 1 tutte le volte che si verifica una interruzione di video.

Forse sarà meglio chiarire l'utilizzo di questo registro con un esempio diretto, anche perché è necessario compiere alcune operazioni addizionali durante e dopo la lettura della locazione 53273 (V+25).

Il primo passo: le abilitazioni

Supponiamo di aver già scritto una appropriata routine di raster, di averla allocata (sempre per esempio) da 49152 (C000) in su.

Per attivarla bisogna anche modificare altri registri (i soliti vettori di interruzione, cioè 788 e 789, esa 314 315; i registri del raster, cioè 53266 e bit più significativo di 53265, esa D012 e D011. Il registro di abilitazione del raster, 53274, esa D01A).

Questo semplice programma in BASIC esegue tutte le operazioni iniziali.

```
10 POKE56333,127:REM DISABILITA
GLI INTERRUPT
20 POKE788,0:POKE789,192:REM CAM-
BIA IL VETTORE DI INTERRUZIONE
30 POKE53265, PEEK (53265) AND127:
REM AZZERA IL BIT PIU' SIGNIFI-
CATIVO DEL RASTER
40POKE53266,51:REM SETTA IL PRI-
MO INTERRUPT ALLA 51.ma LINEA
DI SCHERMO
50POKE56333,129:REM RIABILITA LE
INTERRUZIONI
60POKE53274,129:REM ABILITA GLI
INTERRUPT DEL RASTER
```

La linea 20 cambia i valori del vettore di interrupt spostandolo sulla nostra routine.

Le linee 30 e 40 predispongono la prima interruzione alla 51.ma linea di schermo (il bit più significativo del raster viene azzerato).

Ad ogni modo, ricordatevi in una procedura di inizializzazione di settare SEMPRE entrambi i registri (53266 e bit più significativo di 53265). Inoltre, come vedremo più avanti, tutte le volte che il raster raggiunge la posizione prestabilita e fa scattare l'interruzione, deve essere settato sulla interruzione successiva (se avete bisogno di più di un interrupt di video); questa operazione deve essere fatta dalle routines che gestiscono l'interruzione.

La linea 60 abilita le interruzioni del raster ponendo a 1 il bit 0 ed il bit 7 di 53274.

Come si scrivono le routine di raster

Avrete ormai tutti capito che il raster register si gestisce principalmente con procedure in interrupt; procedure non molto diverse da quelle che avete già visto operare o che avete già scritto.
Considerate il semplice programma pubblicato che cambia il colore dello sfondo circa ad un terzo dello schermo video.

Noterete che la prima istruzione carica nell'accumulatore il contenuto della locazione 53273. Se avete letto bene la prima parte dell'articolo, saprete che questa locazione contiene i flag delle interruzioni del circuito video. Il fatto che sia la prima ad essere "controllata" dopo che il processore ha avviato la procedura di interruzione è di fondamentale importanza. Dicevamo prima che le sorgenti di interruzione sul 64 sono parrecchie, anche se normalmente lavorano solo quelle del TIMER A, che permettono la scansione della tastiera.

Essendo quindi necessario sapere quale dispositivo abbia fatto scattare l'interrupt, bisogna verificare, quando entrano in gioco le interruzioni del raster, se il bit meno significativo della locazione 53273 è uguale a uno.

Se questa condizione è vera significa che l'interrupt è scattato a causa del circuito video. Altrimenti bisogna far eseguire al computer la normale procedura di interruzione, che è la scansione della tastiera; la qual cosa può dare dei problemi quando si lavora con il raster.

Gli interrupt che fanno scandire la tastiera possono infatti talvolta sovrapporsi a quelli del raster, e viceversa provocando sfarfallamenti del video; è dunque preferibile in questi frangenti o disabilitare le interruzioni del TIMER A, oppure far scandire la tastiera all'interno di una interruzione video.

Non è finita. Avrete tutti notato che, se l'interruzione è stata effettivamente di raster, ovvero l'AND logico del contenuto della 53273 con 1 è stato soddisfatto, quest'ultimo (il risultato dell'AND) viene scaricato di nuovo nella 53273. È di fondamentale importanza che la cosa venga fatta, poiché il flag di interruzione (che è appunto il 53273) deve essere resettato sempre dopo una interruzione del circuito video; bisogna cioè azzerare il bit più significativo.

Nella linea 50 risiede appunto la fondamentale operazione di cui si parlava prima. Il contenuto dell'accumulatore infatti, se il programma è arrivato sin qui, è 1. Il fatto che venga "storato" in 53273 (esa D019) ha l'effetto di azzerarne il bit più significativo, la qual cosa resetta il registro e non ostacola l'interruzione di video seguente.

Nella linea 60 viene caricato in accumulatore il valore attuale del raster. Se quest'ultimo è minore di 128 (cioè è positivo in complemento a due), viene eseguita la prima delle due procedure (quella dalla linea 100 fino alla 200). Per maggiori chiarimenti leggette la routine disassemblata.

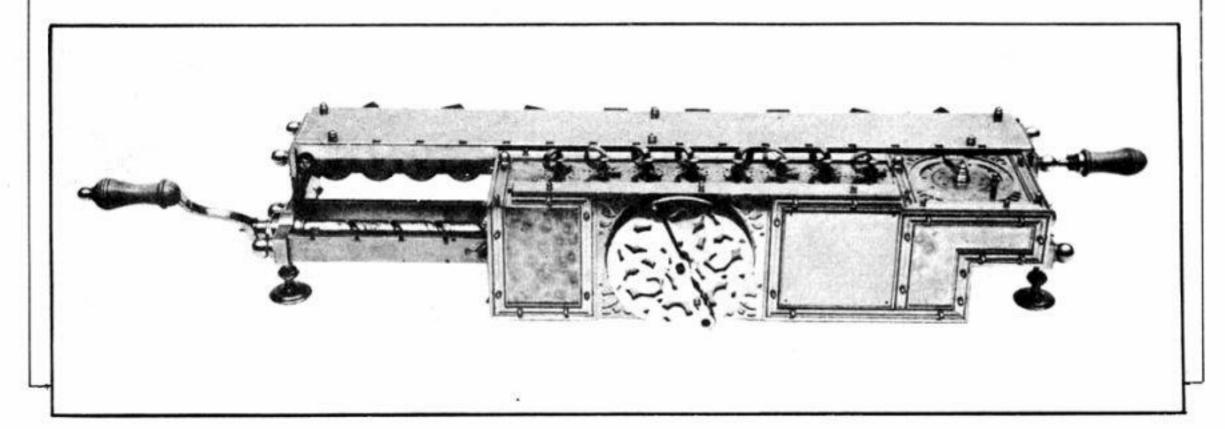
Gli scrolling con il raster register

Forse non tutti sanno che il Commodore 64 permette di scrollare la pagina di testo (notare bene, la pagina di testo, non di alta risoluzione) un Pixel per una volta, sia in senso orizzontale che in senso verticale.

Sto parlando del cosidetto scrolling rallentato che si ottiene "smanettando" i tre bit meno significativi della locazione 53270 (V+22, esa $D016) per variare la posizione x dello schermo ed i tre bit meno significativi di 53265 (v+17, esa $D011) per variare la posizione y dello schermo. Questa opzione è molto comoda per ottenere effetti di animazione, o di scorrimento di messaggi.

Ha una limitazione (in condizioni normali): non può far scrollare parti del video, ma il video intero. Ed ecco che il nostro super raster vi viene in aiuto. Infatti è possibile organizzare scrollate Pixel a Pixel della pagina di testo indipendenti le une dalle altre, ottenendo gli effetti spettacolari che affollano i giochini sul 64.

LE IMMAGINI DI QUESTO FASCICOLO

Fig. 4: La macchina calcolatrice di Leibniz, elaborata nel 1671, perfeziona il meccanismo della Pascalina. Il problema delle moltiplicazioni e delle sottrazioni venne risolto con ripetute addizioni e sottrazioni.

Il principio di funzionamento di queste routine è in parte simile a quello visto prima per dividere lo sfondo.

Le regole fondamentali di cui bisogna tener conto sono le seguenti:

- **sapere esattamente in quante parti bisogna dividere lo schermo, e calcolare tutte le linee del raster, vale a dire tutte le posizioni in cui il raster deve far scattare l'interrupt.**
- **rispetto alle decisioni fatte nel primo punto, locazioni che conservino i valori dello schermo a seconda delle posizioni che il raster register raggiunge.**

Ad esempio: supponiamo di voler dividere lo schermo in due parti, di avere già scritto la nostra routine e di avere utilizzato la locazione decimale 251 per conservare la posizione x della prima parte di schermo, e la locazione 252 per conservare la posizione x della seconda parte di schermo. Di volta in volta la nostra routine, a seconda che il raster abbia fatto scattare l'interruzione nella prima o nella seconda parte del video, scriverà nei tre bit bassi di 53270 (V+22) il valore contenuto in 251 o 252. L'effetto di animazione risiede in un cambiamento del valore contenuto nelle due suddette locazioni, che funzionano un pò come delle variabili in BASIC (cambiamento che in questi casi di solito viene effettuato in interrupt, vedremo come).

- **tenere conto delle imprecisioni del RASTER. Noterete talvolta che tra due aree di schermo l'una indipendente dall'altra i movimenti risultano irregolari. Questo perché la linea puntata dal raster in queste posizioni compone l'immagine della parte superiore e della parte inferiore, cosicché i caratteri visualizzati in quei punti sono imperfetti. L'inconveniente si elimina facilmente, se si fa in modo che i caratteri visualizzati in queste posizioni abbiano rispettivamente parte bassa e parte alta uguali.**

Bisognerà inoltre, in caso di scrolling complessi, tenere conto di quando ogni parte dello schermo raggiunge la massima posizione ammissibile con lo scroll rallentato. Utilizzando infatti tre bit per ottene-

```
10 LDA $D019  ;carica in accumulatore il contenuto di 53273
 20 AND ##$1   ;fa l'and con 1
 30 BNE GO 50  ;se il confronto e' soddisfatto salta alla routine
di raster
 40 JMP $EA31  ;altrimenti salta alla scansione della tastiera
 50 STA $D019  ;azzera il flag di interruzione del video
 60 LDA $D012   ;carica  in  accumulatore  l'attuale  valore  del
raster
 70 BMI GO300  ;se e' maggiore di 127 allora salta alla linea 300
100 LDA ##$80  ;carica in accumulatore il numero 128
110  STA  $D012   ;lo  scrive  nel  raster  register(per  settare
l'interrupt seguente)
120  LDA  $FC      ;carica  in  accumulatore  il  contenuto  della
locazione 252
130 STA $D021  ;lo stora in 53281 (settta lo sfondo)
140 JMP $EA31  ;uscita dall'interruzione(sansione  della  tastier
sotto l' interruzione di video)
200 NOP
210 NOP
300 LDA ##$20  ;carica in accumulatore il numero 32
310 STA $D012  ;dispone il prossimo interrupt alla  32  linea  di
schermo
320  LDA  $FD      ;carica  in  accumulatore  il  contenuto  della
locazione 253)
330 STA $D021  ;lo stora in 53281 (setta lo sfondo)
340 JMP $FEBC  ;esce dall'interrupt
```

```
100 REM  *** STUDIO DEL RASTER REGI
    STER ***
110 REM  ***   MODIFICA DEL COLORE
    DEL  ***
120 REM  ***     FONDO DELLO SCHERM
    O    ***
130 REM  ***
         ***
140 REM  ***     ENRICO SCELSA
         ***
150 REM
160 DATA  173,25,208,41,1,208,3,76,
    49,234
170 DATA  141,25,208,173,18,208,48,15,165
180 DATA  252,141,33,208,169,128,141,18,208
190 DATA  76,49,234,234,234,165,253,141,33,208
200 DATA  169,2,141,18,208,76,188,254,0
210 FOR X=49152 TO 49198:READ A:POKE X,A:NEXT
220 INPUT "COLORE PARTE 1";A:IF A<0 OR A>255
    THEN 220
230 INPUT "COLORE PARTE 2";B:IF B<0 OR B>255
    THEN 230
240 POKE 252,A:POKE 253,B
250 POKE 56333,127:POKE 788,0:POKE 789,192:POKE
    53274,129
260 POKE 53265,PEEK(53265) AND 127:POKE 53266,40
270 END
```

re lo spostamento, vale a dire un numero da 0 a 7, non appena il registro in questione conterrà 8 nel caso che si scrolli a destra (lo schermo si sposta gradatamente verso destra al crescere del contenuto dei tre bit bassi della locazione 53270) dovrà essere ricaricato col valore 0 (la posizione più a sinistra raggiungibile col solo scrolling di un pixel per volta) e si dovrà eseguire lo scrolling verso destra di un carattere.

Viceversa, nel caso dello scrolling a sinistra, non appena il contenuto del suddetto registro, decrementato, passa da 0 a 255, dovrà essere riscritto col valore 7 (che è la posizione più a destra ammissibile con il solo scrolling di un pixel alla volta) e bisognerà eseguire uno scroll di un carattere verso sinistra dell'area video in questione).

Esaminiamo la routine

Questo è il momento di vedere direttamente come funzione una routine del raster che esegue gli scroll.

```
  10 LDA $D019 ;CARICA IL FLAG DI INTERRUPT (DEC 53273)
  20 AND #$1   ;CONTROLLA SE E'DI RASTER
  30 BNE GO50  ;SI VAI A CINQUANTA
  40 JMP $EA31 ;NO VAI ALLA SCANSIONE DI TASTIERA
  50 STA $D019 ;RESETTA IL FLAG DI INTERRUZIONE
  60 LDA $D012 ;CARICA LA POSIZIONE DEL RASTER (DEC 53266)
  80 CMP #$51  ;LA COMPARA CON 81 DECIMALE
  90 BCS GO200 ;SE E' MAGGIORE O UGUALE VAI ALLA LINEA 200
 100 LDA $D016 ;CARICA IL CONTENUTO DI 53270
 110 AND #$F8  ;AZZERA LA POSIZIONE X DELLO SCHERMO
 120 ORA $FC   ;SCRIVE QUELLA DELLA PARTE CHE IL RASTER STA' TRACCIANDO
 130 STA $D016 ;LA STORA IN 53270
 140 LDA #$51  ;CARICA IL PROSSIMO  INTERRUPT DI RASTER
 150 STA $D012 ;LO PREDISPONE
 160 JMP $EA31 ;SCANDISCE LA TASTIERA
 170 NOP       ;****SECONDA PARTE
 180 NOP       ;****    DI
 190 NOP       ;****SCHERMO
 200 CMP #$71  ;COMPARA CON 113 DECIMALE
 210 BCS GO500 ;SE E' MAGGIORE O UGUALE VAI A 500
 220 LDA $D016 ;CARICA IL CAONTENUTO DI 53270
 230 AND #$F8  ;AZZERA I TRE BIT MENO SIGNIFICATIVI (LA X DELLO SCHERMO)
 240 ORA $FD   ;SETTA LA NUOVA X DELLO SCHERMO
 250 STA $D016 ;LA STORA IN 53270
 260 LDA #$71  ;PROSSIMO INTERRUPT ALLA
 270 STA $D012 ;113.MA LINEA DELLO SCHERMO
 280 JMP $FEBC ;ESCE DALL'INTERRUPT
 290 NOP       ;****TERZA PARTE
 300 NOP       ;**** DI
 310 NOP       ;****SCHERMO
 500 LDA #$30  ;PROSSIMA INTERRUZIONE DI VIDEO
 510 STA $D012 ;ALLA 48.MA LINEA
 520 LDA #$F8  ;AZZERA LA POSIZIONE X DELLO SCHERMO
 530 AND $D016 ;
 540 STA $D016 ;
 550 JSR GO2000;SUBROUTINE DI SCROLLING DELLA PRIMA PARTE DI SCHERMO
 560 JSR GO3000;SUBROUTINE DI SCROLLING DELLA SECONDA PARTE DI SCHERMO
 570 JMP $FEBC ;ESCE DALL'INTERRUPT
 580 NOP       ;
 590 NOP       ;
 600 NOP       ;
2000 INC $FC   ;SPOSTA A DESTRA LA PRIMA PARTE DI SCHERMO
2007 LDA 252
2010 CMP #8    ;
2020 BCC GO2060 ;NO ESCE
2030 LDA #0    ;ALTRIMENTI RIPRISTINA I VALORI NORMALI
2040 STA $FC   ;
2050 JMP GO8000 ;E SCROLLA DI UN CARATTERE
2060 RTS       ;
2070 NOP         ;*******SECONDA PARTE
2080 NOP         ;*******     DI
2090 NOP         ;*******SCHERMO
```

```
3000 DEC $FD    ;SPOSTA A SINISTRA LA SECONDA PARTE DI SCHERMO
3020 BPL GO3060 ;NON SUPERA IL LIMITE?NO ESCE
3030 LDA #7     ;ALTRIMENTI RIMETTE LO SCHERMO A DESTRA
3040 STA $FD    ;
3050 JMP GO9000 ;E SCROLLA DI UN CARATTERE
3060 RTS        ;
3070 NOP        ;
3080 NOP        ;
3090 NOP        ;
8000 LDX ##9F   ;SCROLLA LA PRIMA PARTE DI SCHERMO
8010 LDA $03FF;X ;
8020 STA $0400;X ;
8030 DEX        ;
8040 BNE GO8010 ;
8042 NOP        ;QUESTA SUB PASSA TRAMITE L'ACCUMULATORE
8050 JSR  GO9200 ;IL CODICE DI SCHERMO DEL PROSSIMO CARATTERE DA VISUALIZZARE
8060 STA $0400  ;QUI IL CODICE VIENE VISUALIZZATO
8070 STA $0428  ;
8080 STA $0450  ;
8090 STA $0478  ;
8100 RTS         ;ESCE
8110 NOP
8120 NOP
8130 NOP
9000 LDX #0
9005 LDY ##A0   ;SCROLLA LA SECONDA PARTE DI SCHERMO
9010 LDA $04A1;X ;
9020 STA $04A0;X ;
9030 INX        ;
9035 DEY        ;
9040 BNE GO9010 ;
9042 NOP        ;QUESTA SUB PASSA I CODICI DI SCHERMO
9050 JSR GO9300 ;DEI CARATTERI DA VISUALIZZARE
9060 STA $04C7  ;QUI VENGONO VISUALIZZATI
9070 STA $04EF  ;
9080 STA $0517  ;
9090 STA $053F  ;
9100 RTS        ;
9110 NOP
9120 NOP
9130 NOP
9200 LDY $F7    ;INDICE DEL CARATTERE DA PRELEVARE
9210 LDA $8000;Y ;LO PRELEVA
9220 DEC $F7    ;DECREMENTA L'INDICE
9230 BNE GO9260 ;SE NON HA TERMINATO ESCE
9240 LDY $F9    ;ALTRIMENTI RIPRISTINA
9250 STY $F7   ;L'INDICE
9260 RTS         ;
9270 NOP         ;****QUADRO
9280 NOP         ;****DUE
9290 NOP         ;
9300 LDY $F8    ;INDICE CARATTER DA PRELEVARE
9310 LDA $8000;Y ;LO PRELEVA
9320 INC $F8    ;INCREMENTA L'INDICE
9325 CPY $F9    ;E VEDE SE HA TERMINATO DI PRELEVARE CARATTERI
9330 BCC GO9360 ;NO ESCE
9340 LDY ##1    ;ALTRIMENTI RIPRISTINA
9350 STY 248    ;L'INDICE
9360 RTS         ;
9999 END
```

```
ESPERIMENTI COL RASTER REGISTER (2)
100 REM       STUDIO DEL RASTER REGI
    STER:
110 REM  SCROLL DI STRINGHE IN SENS
    O INVERSO
120 REM              ENRICO SCELSA
130 REM
140 DATA  173,25,208,41,1,208,3,76,
    49,234
150 DATA  141,25,208,173,18,208,201
    ,81,176
160 DATA  21,173,22,208,41,248,5,25
    2,141,22
170 DATA  208,169,81,141,18,208,76,
    49,234
180 DATA  234,234,234,201,113,176,2
    1,173,22
190 DATA  208,41,248,5,253,141,22,2
    08,169,113
200 DATA  141,18,208,76,188,254,234
    ,234
210 DATA  234,169,48,141,18,208,169
    ,248,45,22,208
220 DATA  141,22,208,32,91,196,32,1
    10,196
230 DATA  76,188,254,234,234,234,23
    0,252,165,252
240 DATA  201,8,144,7,169,0,133,252
    ,32,125
250 DATA  196,96,234,234,234,198,25
    3,16,7
260 DATA  169,7,133,253,32,156,196,
    96,234
270 DATA  234,234,162,159,189,255,3
    ,157,0,4
280 DATA  202,208,247,234,32,190,19
    6,141,0
290 DATA  4,141,40,4,141,80,4,141,1
    20,4
300 DATA  96,234,234,234,162,0,160,
    160,189
310 DATA  161,4,157,160,4,232,136,2
    08,246,234
320 DATA  32,207,196,141,199,4,141,
    239,4
330 DATA  141,23,5,141,63,5,96,234,
    234,234
340 DATA  164,247,185,0,128,198,247
    ,208,4
350 DATA  164,249,132,247,96,234,23
    4,234,164,248
360 DATA  185,0,128,230,248,196,249
370 DATA  144,4,160,1,132,248,96,0,
380 REM
390 FOR X=50176 TO 50399:READ A:POK
    E X,A:NEXT
400 PRINT"[CLEAR]"
410 PRINT"[5 DOWN][3 RIGHT] INTRODU
    RRE LA STRINGA DA SCROLLARE"
420 INPUT  A$
430 PRINT"[CLEAR] ";A$:POKE 249,LEN
    (A$)+1:POKE 248,1:POKE 247,1:
440 FOR X=1 TO LEN(A$)+1:POKE 32768
    +X,PEEK(1023+X):NEXT:PRINT"[CLE
    AR]"
450 POKE 56333,127:POKE 788,0:POKE
    789,196
460 POKE 53274,129:POKE 53265,PEEK(
    53265) AND 127:POKE 53266,40
470 POKE 53281,0
480 PRINT"[15 DOWN]"
490 GOTO 490
```

LE IMMAGINI DI QUESTO FASCICOLO

Fig. 5: Ecco la prima scheda perforata. Venne creata nel 1804 dal francese Joseph Marie Jacquard per controllare il movimento dei telai da tessitura. Attraverso i fori, degli uncini scendono ad afferrare i fili del tessuto.

BREAK OUT

Super gioco per Vic 20 inespanso

Di questo gioco esistono molte versioni.

Nuovissimo è invece il modo di utilizzare il popolare computer della Commodore che viene qui chiamato a modificare addirittura il dimensionamento righe-colonne dello schermo.

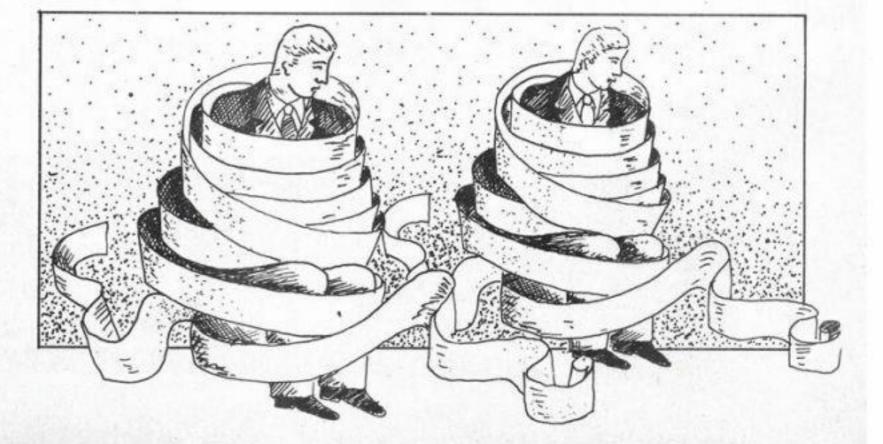
Come era stato preannunciato sull'articolo che studiava il circuito integrato 6561 (N.17 CCC), ecco che pubblichiamo un gioco fondato sull'impiego razionale delle sue numerose locazioni di memoria.

Il gioco è scritto interamente in Linguaggio Macchina ma viene allocato in memoria mediante istruzioni READ e POKE di due programmi Basic. La tecnica di caricamento avviene in overlay (vedi CCC N.9).

Ciò vuol dire che è necessario attenersi scrupolosamente alle regole che seguono:

- Digitare il primo listato (il più lungo) registrarlo e verificarlo SENZA farlo girare e solo dopo averlo scrupolosamente controllato.
- Digitare il secondo listato, registrarlo e verificarlo in modo che sul nastro si trovi SUBITO DOPO quello precedente.
- Tornare indietro col nastro, caricare il primo listato e dare il RUN lasciando abbassato il tasto play del registratore.
- Dopo alcuni secondi il secondo programma sarà automaticamente caricato (riga 10 primo listato) e lanciato premendo il tasto F1.
- Se avete digitato i due programmi correttamente lo schermo del Vic diventerà stretto e alto e la velocità del gioco (interamente in linguaggio macchina) darà del filo da torcere anche ai più bravi!

La pressione della barra spaziatrice fa partire una delle tre palline a disposizione mentre la racchetta si sposta a destra o a sinistra mediante i due tasti cursore.

G. Puricelli

BREAK OUT (1)

```
 5 PRINT"[CLEAR]":POKE 52,25:POKE
   56,25
10 FOR I=6400 TO 6993:READ A:POKE
   I,A:NEXT:LOAD
20 DATA 162,0,189,0,128,157,0,28,2
   32,224,0,208,245,238,4,25,238,7
   ,25,173,7,25
21 DATA 201,30,208,230,189,161,2,1
   57,0,28,232,224,40,208,245,169,
   14,141,15,144,169,255
22 DATA 141,5,144,169,22,141,1,144
   ,169,190,141,3,144,169,18,141,0
   ,144,169,144,141,2
23 DATA 144,165,197,201,39,208,250
   ,162,0,32,226,26,76,128,27,32,9
   4,27,189,0,30,74
24 DATA 176,5,169,32,157,0,30,24,1
   65,87,101,88,170,165,91,157,0,3
   0,173,105,25,41
25 DATA 31,141,86,25,141,94,25,201
   ,30,208,81,165,96,74,176,39,138
   ,74,189,1,30,144
26 DATA 3,189,255,29,74,144,26,138
   ,74,169,32,144,5,157,255,29,176
   ,3,157,1,30,169
27 DATA 1,133,96,32,16,27,234,234,
   234,234,234,165,90,74,176,16,16
   5,88,201,16,208,86
28 DATA 169,1,133,92,32,87,27,76,1
   4,26,165,88,201,240,208,78,169,
   0,133,88,238,105
29 DATA 25,76,21,26,165,90,74,176,
   16,165,88,201,0,208,49,206,105,
   25,169,240,133,88
30 DATA 76,21,26,165,88,201,192,24
   0,7,201,240,208,37,76,150,27,18
   9,16,31,201,248,208
31 DATA 27,165,91,201,2,208,10,169
   ,0,197,89,208,2,169,1,133,89,32
   ,87,27,56,165
32 DATA 88,233,16,76,19,26,24,165,
   88,105,16,133,88,165,89,74,176,
   32,165,91,201,0
33 DATA 208,18,165,87,201,0,208,5,
   169,1,76,72,26,198,87,169,4,76,
   101,26,74,201
34 DATA 1,240,251,76,101,26,165,91
   ,201,4,208,27,165,87,201,15,208
   ,14,169,0,133,89
35 DATA 169,220,141,12,144,169,2,7
   6,101,26,230,87,169,0,76,101,26
   ,201,0,208,5,169
36 DATA 2,76,101,26,10,133,91,173,
   139,26,201,14,240,19,165,197,20
   1,23,208,6,238,139
37 DATA 26,76,138,26,173,139,26,20
   1,0,240,9,165,197,201,31,208,3,
   206,139,26,160,8
38 DATA 169,248,153,208,31,165,92,
   74,169,32,153,207,31,153,210,31
   ,176,2,169,248,153,209
39 DATA 31,144,5,169,15,141,107,26
   ,134,93,162,0,189,64,30,74,176,
   20,232,224,128,208
40 DATA 245,165,90,201,1,208,9,165
   ,88,201,224,208,3,32,226,26,166
   ,93,169,0,133,94
41 DATA 133,95,230,94,208,252,230,
   95,165,95,201,32,208,244,141,12
   ,144,76,81,25,160,0
42 DATA 169,7,153,64,150,169,3,153
   ,96,150,169,2,153,128,150,169,5
   ,153,160,150,200,192
43 DATA 32,208,231,160,0,169,1,153
   ,64,30,169,3,153,65,30,200,200,
   192,128,208,240,96
44 DATA 134,93,162,5,189,0,30,201,
   57,240,6,254,0,30,76,43,27,169,
   48,157,0,30
45 DATA 202,224,1,208,233,173,3,30
   ,201,49,208,6,238,12,30,238,47,
   27,166,93,165,88
46 DATA 201,128,16,7,234,234,169,2
   1,141,217,26,169,230,141,12,144
   ,169,1,197,90,208,2
```

BREAK OUT (2)

```
 1 REM  ** BREAK-OUT **
 2 REM  ***************
 3 REM  BY G. PURICELLI
10 FOR I=6994 TO 7129:READ A:POKE
   I,A:NEXT
12 FOR I=673 TO 712:READ A:POKE I,
   A:NEXT:POKE 36878,15
15 PRINT"[CLEAR][GIALLO][HOME][6 D
   OWN]PUSH F1":SYS6400
20 FOR U=0 TO 3000:NEXT:PRINT"[CLE
```

```
   AR]":POKE 36869,240:PRINT"[GIAL
   LO][HOME][8 DOWN][4 RIGHT]GAME
   OVER"
21 PRINT"[DOWN][11 LEFT]PUSH F1 TO
    PLAY"
22 IF PEEK(197)=39 THEN PRINT"[CLE
   AR]":GOTO 25
24 GOTO 22
25 POKE 36869,255:SYS6473
30 GOTO 20
50 DATA 169,0,133,90,96,169,0,133,
   96,76,71,27,134,93,162,0,24,189
   ,64,30,125,65
51 DATA 30,234,74,144,10,169,1,157
   ,64,30,169,3,157,65,30,232,232,
   224,128,208,229,166
52 DATA 93,96,160,0,169,48,153,1,3
   0,200,192,6,208,248,169,51,141,
   12,30,169,49,141
53 DATA 47,27,173,12,30,201,48,208
   ,1,96,206,12,30,165,197,201,32,
   208,250,169,192,133
54 DATA 88,169,1,133,89,133,90,169
   ,0,133,91,133,92,133,96,169,32,
   141,217,26,169,14
55 DATA 141,107,26,169,30,141,105,
   25,141,86,25,141,94,25,32,148,2
   24,165,140,41,13,133
56 DATA 87,76,81,25
60 DATA 0,0,240,240,240,240,240,0,
   0,127,127,127,127,127,127,0,0,0
   ,60,60,60,60
61 DATA 60,0,0,254,254,254,254,254
   ,254,0,0,0,15,15,15,15,15,0
```

FIG 7

FIG. 8

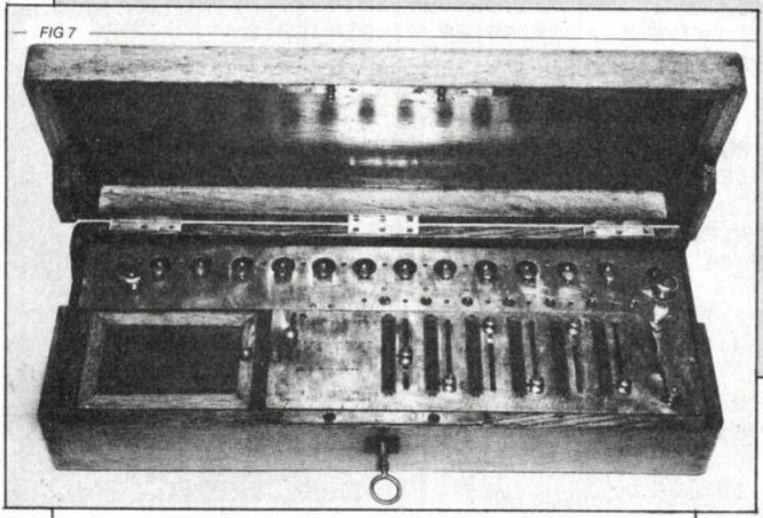
FIG. 9

LE IMMAGINI DI QUESTO FASCICOLO

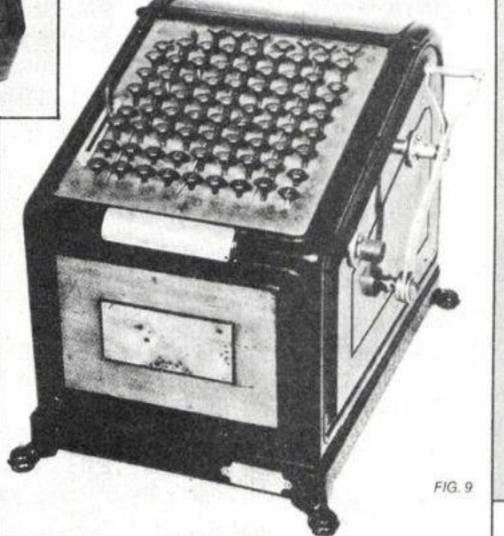
FIG. 7

Il primo grande successo commerciale nella storia dell'elaborazione dei dati: l'aritmometro del finanziere francese Charles Xavier Thomas de Colmar. Migliaia di esemplari venduti dal 1820 al 1890.

FIG. 8

Il problema di effettuare le moltiplicazioni viene risolto. La macchina del diciotenne Leon Bollee non utilizza più il sistema delle addizioni ripetute. Aste di lunghezza diseguale (nove linee e nove colonne) sono fissate su alcune piastre metalliche. La moltiplicazione viene effettuata meccanicamente. E' il 1887.

FIG. 9

Un'altra macchina a moltiplicazione diretta. Quella di William S. Burroughs.

GHOSTBUSTHERS

In italiano significa "acchiappafantasmi" ed è proprio questo lo scopo del videogioco che ricalca fedelmente la trama del film ononimo e la canzone di Ray Parker Junior.

I protagonisti: tre simpatici ragazzi che fondono una società per aiutare i cittadini newyorkesi a liberarsi dai fantasmi. Proprio questo paradosso, il fatto che allegramente dei ragazzi affrontino misteri dell'oltretomba, ha permesso al film di ottenere un successo strepitoso sia in America che in Italia incassando milioni di dollari. I fantasmi si riuniscono tutti in una centrale dove stanno preparando strategie e diabolici piani per distruggere gli esseri umani. Lo scopo è quello di raggiungere il tempio di ZUUL, la centrale, dopo aver catturato il maggior numero di fantasmi e soldi.

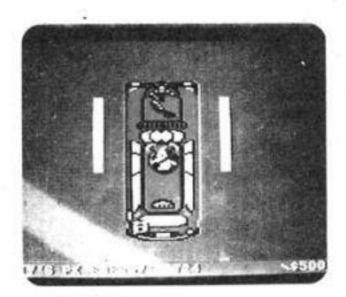
Il programmatore di questo gioco, il bravissimo David Crane, (autore di bellissimi giochi dell'Activision tra cui Decathlon), è riuscito a realizzare nel gioco le stesse reali situazioni del film grazie all'ottima grafica ed alla stupenda musica (fedelissima alla colonna sonora). Se al tutto aggiungiamo l'urlo iniziale con la risata spettrale non possiamo fare a meno di giudicarlo uno dei migliori computer game per il C64.

La presentazione del gioco è veramente bella: riduce il simbolo di Ghostbusthers assieme all'urlo con la risata spettrale che prima dicevamo. Un messaggio scritto lettera per lettera vi chiede il vostro nome e se avete un numero di conto. Se non siete mai riusciti a sconfiggere i fantasmi non avrete mai un vostro numero di conto e quindi la banca vi presenterà, per iniziare la vostra società, 10000 dollari. Vi serviranno per acquistare l'automobile e l'attrezzatura.

L'automobile potrà essere scelta tra quattro tipi, che differiscono tra loro per velocità di crociera e numero di attrezzi trasportabili. Il prezzo varia da un minimo di 2000 dollari per l'utilitaria ad un massimo di 15000 dollari per la sportiva. La velocità è molto importante, perchè meno tempo impiegate per percorrere le strade della città meno veloci saranno i Roamers, i fantasmi che si aggirano per le strade.

L'attrezzatura disponibile è suddivisa in tre schermate differenti. Nella prima troviamo il rivelatore di energia PK, che rivela l'arrivo di uno Slimer, fantasma che abita nelle case: cambia in rosa il colore di un edificio e costa 400 $. L'intensificatore di immagini che vi permette di vedere e quindi catturare meglio gli Slimers, costa 800 $. Infine il sensore per il Marshmallow, il mostruoso fantasma che si forma quando un numero sufficiente di Roamers ha raggiunto il tempio di ZUUL, che vi avverte del suo arrivo cambiando il colore dell'edificio in bianco e costa 800 $.

Nella seconda schermata troverete l'aspirafantasmi che costa 500 $ e vi aiuta ad aspirare i Roamers che avete incontrato durante il vostro pattugliamento per le strade della città. Le trappole per catturare i fantasmi costano 600 $ l'una e contengono un solo fantasma, quindi è con-

veniente comprare almeno due per non dovere tornare ogni volta al quartier generale, GHQ, dopo aver catturato un fantasma. Infine per 400 $ vi offrono l'esca attira Roamers che vi permette di guadagnare migliaia di dollari distruggendo il Marshmallow tramite il tasto B, prima che distrugga l'edificio, in questo caso le migliaia di dollari le perderete voi.

Nell'ultima schermata c'è un sistema che permette di contenere ben cinque Slimers ma costa 8000 $. Lo comprerete molto più avanti!!!

A questo punto vi verrà presentata la mappa della città con l'energia PK che sale progressivamente e i dollari che possedete. Quando un edificio diventa rosso significa che al suo interno c'è uno Slimers; con il joystick dirigetevi verso di esso cercando di intercettare più Roamers possibili poichè ognuno di essi ogni volta che raggiunge la centrale aumenta di 100 l'energia PK. Quando siete a fianco dell'edificio premete il pulsante del joystick ed entrerete nel secondo quadro: viaggiando su una larga strada aspirerete i Roamers che avete intercettato. Arrivati alla casa posizionate la trappola ed il primo gostbusthers girandolo verso la trappola e premete di nuovo il pulsante. Ora arriverà il secondo ghostbusthers, dopo averlo posizionato premete il pulsante per attivare i laser. Quando lo Slimer sarà al centro dei due raggi premete di nuovo il pulsante per intrappolarlo.

Nel caso di vittoria guadagnerete tanti dollari più sarete stati veloci nella cattu-

giochiamo con...

ra; in caso di fallimento perderete un ghostbusters e l'energia PK aumenterà di 300; incrociando i raggi perderete tutti e due gli uomini.

Il joystick vi permette di scegliere in quale edificio entrare mentre la barra spaziatrice permette durante la presentazione di far ripetere la parola ghostbusthers e durante il gioco dà la situazione della vostra energia, trappole, uomini. Quando vi troverete con meno di due uomini o con trappole vuote dovrete tornare al quartier generale.

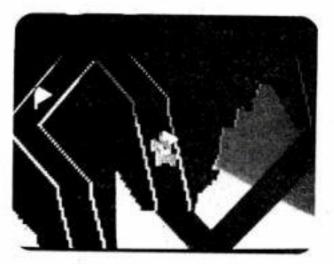
La fase finale del gioco inizia quando la chiave e il lucchetto si avvicinano al tempio di ZUUL e vi entrano. A questo punto il gioco può finire in due modi diversi. Se vi trovate con meno dollari di quanti ve ne ha prestati la banca avete già perso altrimenti la banca vi comunicherà il numero di conto corrente che vi permetterà nel futuro di fare uso dei dollari guadagnati dando il nome ed il conto corrente associato. Nel caso la banca vi comunichi il conto dovrete cercare di inserire due ghosbusthers all'interno del tempio di ZUUL evitando l'enorme Marshmallow.

A questo punto avete finito i fantasmi e salvato tutti gli abitanti di New York.

Idea:	8
Giocabilità:	7
Grafica e suono:	9
Animazione:	8
Voto:	9

UP 'ND DOWN

Dopo il bellissimo Buck Roger ecco un altro gioco affascinante della casa di software americana. Stavolta non si tratta di guerre spaziali ma di un simpaticissimo viaggio in automobile attraverso paesaggi fantastici.

La prima schermata permette al giocatore di scegliere tra tre livelli di difficoltà: facile (si parte dal primo livello), medio (dal secondo livello), difficile (dal terzo

livello)tra uno o due giocatori, e se controllare la nostra automobile dal joystick o dalla tastiera.

Per iniziare il viaggio bisogna premere il pulsante del joystick o la barra spaziatrice. Ci si presenterà un paesaggio pieno di prati attraversato da alcune strade di cemento che si incrociano tra di loro. In alto sullo schermo ci verranno fornite delle informazioni molto importanti. Partendo da sinistra troveremo il giocatore con il suo relativo punteggio, il massimo punteggio raggiunto, il livello di gioco, le automobili che ci restano da giocare ed infine dieci bandierine colorate (che sono quelle che noi dovremo conquistare).

Il punteggio aumenta con il passare del tempo, con le bandierine conquistate e con le automobili che faremo scoppiare cadendoci sopra. Nel terzo quadro incontreremo delle automobili con una bandierina sul tetto: ce la regaleranno cadendoci sopra.

E' possibile in tutti i quadri saltare da una strada all'altra scavalcando prati o precipizi o il mare. Bisogna anche stare molto attenti alle salite e discese perchè perdendo velocità rischieremo di andare all'indietro schiantandoci contro le automobili che si seguono o di acquistare troppa velocità e di tamponare le automobili che ci precedono.

Automobile in regalo con 10000 punti e ogni 20000 successivi.

Idea:	8
Giocabilità:	8
Grafica e suono:	7.5
Animazione:	7
Voto:	8.5

BOOTY

Nel gioco Booty, prodotto dalla casa di software Firebird e distribuito dalla Mastertronic, il tema avventuroso si fonde perfettamente con quello fantastico.

Il marinaio Jim dovrà dapprima cercare di impossessarsi del maggior numero possibile di oggetti, presenti all'interno della stiva di un galeone pirata.

Per conseguire questo risultato dovrà superare parecchie trappole che lo porterebbero ad una morte crudele, divorato dai topi di bordo o tagliato a fettine da pirati fantasma.

Se riuscirà a superare queste difficoltà, ed a raccogliere tutti gli oggetti presenti, avrà a disposizione 45 secondi per localizzare una chiave di bronzo ed aprirsi la via verso il ponte di coperta del Galeone Nero.

Anche qui, ma con pericoli maggiori, il nostro eroe dovrà collezionare tutti i tesori che trova sul suo percorso, per acquisire una chiave d'argento che gli permetterà di accedere al ponte di comando.

Quando giungerà in questa parte del galeone, l'eroico marinaio Jim dovrà per la terza e ultima volta appropriarsi di tutti gli oggetti che vi si trovano, al fine di ottenere la chiave d'oro che gli permetterà la fuga dalla nave fantasma.

Idea	7
Giocabilità	è
Grafica e suono	8
Animazione	8
Voto	8

BIG-MAC

Il gioco Big-Man della casa di software Mastertronic, si contraddistingue per la varietà e la dinamicità delle situazioni di gioco a disposizione dell'utente.

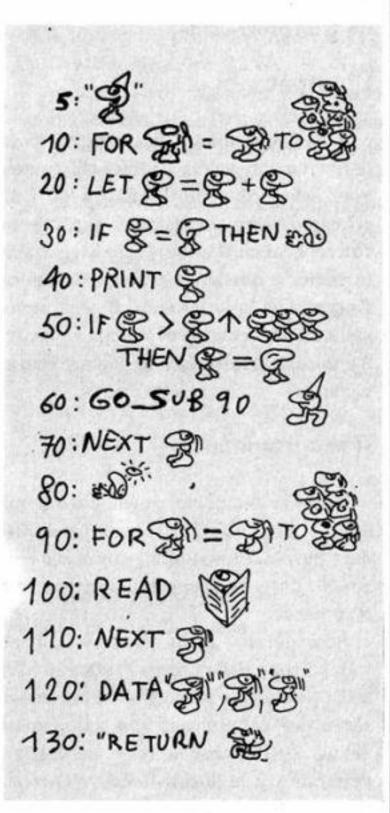
L'agente speciale Big Mac deve cercare di introdursi all'interno di una base segreta nemica, nel tentativo di neutralizzare la produzione di energia elettrica.

Le informazioni che sono pervenute dai servizi di informazione alleati, gli hanno consentito di appurare l'esistenza di 18 diverse centrali interne destinate alla produzione dell'energia.

Quando l'agente segreto Big Man giungerà nella fortezza nemica gli schemi di protezione verranno immediatamente attivati.

Il funzionamento dell'impianto di ossigenazione interno verrà inibito ed avrà a disposizione un tempo limitato per concludere la propria missione di sabotaggio.

L'ottima grafica e l'eccellente sonoro renderanno ancora più avvincenti le situazioni di gioco che progressivamente si susseguiranno.

Idea	8
Giocabilità	8
Grafica e suono	8
Animazione	8
Voto	—

COMMODORE 64 & STAMPANTE MPS 802

TEXT COPY

Copiare la pagina testo su carta utilizzando la stampante MPS 802.

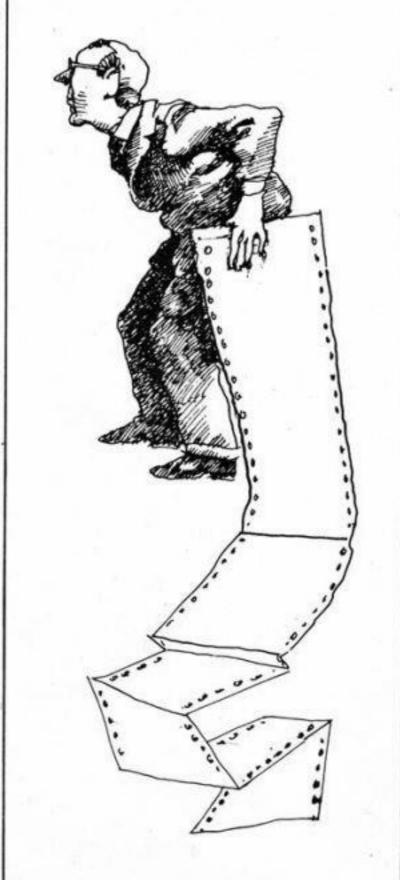
La routine vera e propria inizia dalla linea 30000. Le linee precedenti servono a creare una schermata e a mostrare come inserire ed utilizzare la routine all'interno di un programma.

L'output

Una volta copiato l'intero listato (potete ovviamente saltare le linee che contengono solo i "due punti") date il RUN. Lo schermo verrà riempito con scritte e caratteri grafici. Il risultato, se avete copiato bene, è quello mostrato nella figura. Completata la schermata, il programma salta alla subroutine di stampa e, in circa un minuto, trasferisce la pagina testo su carta.

Il programma

Sulla prima parte, quella che riempie lo schermo, non credo valga la pena spendere parole. Passo quindi subito alla linea 30000, dove viene aperto il canale 6 della stampante.

Ricorderete certo (vedere articolo "Hard copy della pagina grafica su MPS 802" pubblicato sul n. 18) che tale canale serve per comunicare alla stampante il valore dell'interlinea (cioè la distanza verticale tra le linee). Il valore che consente di scrivere le linee una sotto l'altra senza spazi è 21. Se ricordate, nell'articolo sopra citato, dicevo che era 20 il numero da comunicare. Ebbene, nessuno dei due valori è completamente errato o esatto.

La stampante infatti non è precisissima: alcune volte le linee, a parità d'interlinea, risultano perfettamente combacianti, altre volte rimane un piccolo spazio o vengono leggermente a sovrapporsi. Nel caso dell'hard copy della pagina grafica ho preferito il valore 20, che evita nel 99% dei casi gli "scollamenti" (al prezzo di un maggiore numero di sovrapposizioni). In questa routine invece ho optato per l'altra possibilità. Nulla vi impedisce di scegliere l'interlinea che preferite.

Nella linea 30010 vengono aperti due files: il numero 3 per comunicare con lo schermo 4 (che nella linea precedente era stato già usato per altri scopi) per la stampante. Questa viene utilizzata per l'output, mentre il video verrà visto come periferica in input. Il CHR$(19), che riporta il cursore in alto a sinistra, serve per poter leggere il "file-schermo" dall'inizio.

Le linee dalla 30030 si occupano di prendere i dati del file-schermo (GET 3,A$) e trasferirli alla stampante (PRINT 4,...) mantenendo il formato di 40 colonne. Purtroppo i caratteri in reverse non compaiono come tali nel file-schermo ma nella loro versione normale.

TABELLA ISTRUZIONI ASSEMBLER

di Simone Bettola

Ecco la tabella con tutto il set di istruzioni del 6502 che, ormai è noto, è il microprocessore utilizzato dai nostri beneamati Commodore 64, VIC 20 ed ora, nella versione 7501, anche dal Commodore 16 e dal PLUS 4.

E' una guida pratica, comoda, che può essere utilizzata in qualsiasi situazione e per qualsiasi scopo dal principiante e da chi già conosce qual cosa di linguaggio macchina. Ma bando ai preamboli. Entriamo nel merito della tabella.

La tabella

Il set completo di istruzioni utilizzabili dal programmatore in Linguaggio Macchina consta di 56 operazioni diverse. Ciascuna di queste operazioni viene utilizzata sfruttando un determinato indirizzamento in tutto sono disponibili 13 indirizzamenti, che analizzeremo più avanti. Naturalmente non sono permessi tutti i possibili abbinamenti tra operazione ed indirizzamento, quindi, alla resa dei conti, potremo sbizzarrirci con la "modica" cifra di 151 istruzioni vere e proprie.

A scanso di equivoci premetto che in questa tabella è stato volutamente tralasciato un dato caratteristico del funzionamento del microprocessore. Alludo al numero dei "cicli" necessari per ciascuna istruzione. Per dirla all'americana, il "timing".

Questa omissione è stata voluta proprio per motivi di chiarezza.

Nella tabella (per usarla dovrete staccare i due fogli e farli combaciare) ogni operazione interviene in un campo diverso, dando luogo alla vera e propria istruzione. Così verticalmente, nella prima colonna a partire da sinistra, sono riportate le parole chiave a nostra disposizione (ADC,AND,ecc), mentre orizzontalmente, nella prima riga in alto, le varie indicizzazioni abbinabili.

Il punto di incontro della riga con la colonna individuerà un campo, caratteristico dell'istruzione che vogliamo utilizzare. Questo campo contiene in realtà tre valori diversi, che vanno sempre tenuti presenti al momento di programmare.

Le prime due cifre riguardano il codice utilizzato per distinguere quella determinata istruzione. L'unica differenza tra i numeri 5e costituita dalla loro base numerica: il primo è espresso in notazione decimale (≠), mentre il secondo in quella esadecimale ($). Tale accorgimento può sembrare superfluo, ma spesso ci si può trovare a dover utilizzare una base piuttosto che l'altra.

Dunque, il primo valore esprime il codice operativo in notazione decimale, il secondo in esadecimale, ed il terzo riguarda la lunghezza effettiva di ogni istruzione, cioè lo spazio utilizzato in memoria.

Ogni istruzione può richiedere 1, 2 o 3 bytes a seconda dell'indirizzamento utilizzato. Il terzo valore del campo (N) riguarda appunto la sua lunghezza globale è importante ricordare che il numero riportato è comprensivo dell'istruzione stessa. Quindi, quando per esempio si dice che una istruzione è "a due bytes", intendiamo (e non solo noi) che oltre al codice operativo stesso abbiamo un solo byte di lavoro.

Un esempio

Supponiamo di avere l'istruzione JSR $a183 e di volere sapere quali valori vengono effettivamente posti in memoria. Innanzitutto cerchiamo la parola chiave nella colonna verticale, poi procediamo orizzontalmente: l'unico indirizzamento disponibile è quello assoluto (guarda caso). Il campo che abbiamo individuato contiene 3 valori: 032,20,3.

Il primo non ci interessa per il momento, è espresso in decimale; il secondo è il codice operativo che cercavamo, mentre il terzo ci dice che la nostra istruzione richiede altri due bytes per specificare l'indirizzo di partenza della nostra subroutine.

L'indirizzamento assoluto (vedi oltre) vuole che i due bytes seguenti l'OP CODE siano espressi nella forma LB,HB. La nostra istruzione si presenta dunque nella memoria come: "20 83 A1". Se noi volessimo creare una routine in L.M. partendo dal Basic con i soliti READ, DATA e POKE, nelle istruzioni DATA dovremmo digitare i seguenti numeri decimali: "DATA 32,131,161".

Nota: *prestate attenzione alla somiglianza fra la lettera "O" ed il numero "0", che non è stato sbarrato diagonalmente.*

ISTRUZIONE		Immediato			Assoluto			Pagina Zero			Accumulatore			Implicito			(IN
Mnemonico	OPERAZIONE	#	$	N	#	$	N	#	$	N	#	$	N	#	$	N	#
ADC	A + M + C → A,C Addiziona con Carry.	105	69	2	109	6D	3	101	65	2							097
AND	A ∧ M → A And logico fra accumulatore e memoria	041	29	2	045	2D	3	037	25	2							033
ASL	C ← 7 6 5 4 3 2 1 0 ← 0				014	0E	3	006	06	2	010	0A	1				
BCC	Branch on Carry Cleared (C = 0) Salta se carry = 0																
BCS	Branch on Carry Set (C = 1) Salta se Carry = 1																
BEQ	Branch on EQual (Z = 1) Salta se uguale																
BIT	A ∧ M (A non viene modificato) And logico fra accumulatore e memoria				044	2C	3	036	24	2							
BMI	Branch on MInus (n = 1) Salta se negativo																
BNE	Branch on Non Equal (z = 0) Salta se non è uguale																
BPL	Branch on Plus (n = 0) Salta se positivo																
BRK	PC + 2 → S, P → S Contatore di programma e status nello stack													000	00	1	
BVC	Branch on oVerflow Cleared (V = 0) Salta se overflow = 0																
BVS	Branch on oVerflow Set (V = 1) Salta se overflow = 1																
CLC	0 → C Pone a 0 il carry													024	18	1	
CLD	0 → D (lavora in esadecimale) Pone a 0 il modo decimale													216	D8	1	
CLI	0 → I Pone a 0 l'interrupt													088	58	1	
CLV	0 → V Pone a 0 l'overflow													184	B8	1	
CMP	A - M Compara l'accumulatore alla memoria	201	C9	2	205	CD	3	197	C5	2							193
CPX	X - M Compara x alla memoria	224	E0	2	236	EC	3	228	E4	2							
CPY	Y - M Compara y alla memoria	192	C0	2	204	CC	3	196	C4	2							
DEC	M - 1 → M Decrementa la memoria				206	CE	3	198	C6	2							
DEX	X - 1 → X Decrementa X							202	CA	2							
DEY	Y - 1 → Y Decrementa Y							136	88	2							
EOR	A ⊻ M → A EOR logico fra accumulatore e memoria	073	49	2	077	4D	3	069	45	2							065
INC	M + 1 → M Incrementa la memoria				238	EE	3	230	E6	2							
INX	X + 1 → X Incrementa X													232	E8	1	
INY	Y + 1 → Y Incrementa Y													200	C8	1	
JMP	JuMP to new location Salta ad una nuova locazione				076	4C	3										
JSR	Jump SubRoutine Salta ad una sottoprocedura				032	20	3										

Gli indirizzamenti

• **Implicito.** Un' istruzione implicita deriva il suo nome dal fatto che non contiene specificamente l'indirizzo dell'operando su cui opera, ed infatti non richiede niente oltre al codice operativo stesso. Tipicamente viene utilizzato per operazioni che riguardano esclusivamente i registri interni. (Es.: INX= E8)

• **Assoluto.** Abbiamo già visto nell'esempio precedente che tale indirizzamento richiede 3 bytes, dove il primo è il codice operativo stesso, ed i restanti due costituiscono l'indirizzo a 16 bit che specifica l'operando. Tale valore, dal momento che il 6502 è un microprocessore ad 8 bit, cioè tratta solo valori fino a 255, viene codificato nella memoria secondo il consueto ordine LB,HB ,ottenuti grazie alla ancora più consueta formula HB= (N/256) , LB= N—256*HB. (Es.: LD A $A9B0 = AD B0 A9)

• **Immediato.** In questo caso l'operando non è contenuto nella locazione puntata dall'istruzione, ma è costituito dal secondo byte stesso; naturalmente (per la limitazione degli 8 bit) sono impegnati solo due bytes. (Es.: LDA≠ $00 = A9 00)

• **Pagina zero.** Per definizione sono impiegati soltanto due bytes e questo è ben lungi dal costituire una limitazione
si può considerare come una variante dell'indirizzamento assoluto, in quanto in pratica il 6502 pone automaticamente a zero il HB dell'operando, senza quindi che ci sia bisogno di specificarlo. Questa forma impiegata consente non solo un minore spazio impiegato, ma anche, e soprattutto, una maggiore rapidità di esecuzione in virtù dell'ormai famoso timing. Va impiegato tutte le volte che si eseguono operazioni sui registri in pagina zero, che come sappiamo, sono molto importanti per il normale funzionamento del calcolatore.(Es. LDA $FB= A5 FB).

• **Accumulatore.** Riguarda solo alcune istruzioni che interessano l'accumulatore, quindi prevalentemente operazioni di scorrimento. E' simile all'implicito, dal quale si differenzia solo concettualmente. (Es.: LSR A = 4A)

• **Indiretto indicizzato.** Sfrutta l'indice X e serve soprattutto per creare tabelle in pagina zero che possono essere scorse semplicemente modificando il registro X. Per maggiori chiarimenti (senza dubbio necessari soprattutto per i meno esperti), rimando al mio articolo in questo stesso numero.
(Es.: LDA($FB,X) = A1 FB)

• **Indicizzato indiretto.** Utilizza l'indice Y e si differenzia dal precedente non solamente per l'indice utilizzato
infatti serve prevalentemente per gestire tabelle in tutta la memoria. Abbinato oppurtunamente al precedente offre caratteriutiche di flessibilità veramente eccezionali. Anche per questa istruzione consiglio di consultare la quinta parte del corso di L.M.
(Es: LDA ($FB),Y = B1 FB)

• **Assoluto indicizzato in X ed in Y.** Nonostante sia molto più semplice del precedente, offre ugualmente ampie possibilità. Grazie a questo indirizzamento si può modificare l'operando di un'istruzione assolutaa semplicementefacendo variare un indice; naturalmente sono necessari 3 bytes. Supponiamo di avere il seguente programmino:

```
LDX ≠ $05
LDA $A000,X
RTS
```

L'accumulatore sarà caricato con il contenuto della locazione $(A000+05).
(Es. : LDA $A000,X = BD 00 A0)

• **Pagina zero indicizzato in X ed in Y.** Variante del precedente, applica le caratteristiche di indicizzazione alle operazioni in pagina zero; per esso valgono tutte le indicazioni già esposte relative all'indirizzamento in pagina zero. Penso che non sia superfluo rilevare che esiste una fondamentale differenza tra questo indirizzamento e quello indiretto indicizzato (e indicizzato indiretto), insita proprio nell'indirezione.
(Es.: LDA $A1,X = B5 A1)

• **Indiretto.** Puro, non abbinato quindi all'indicizzazione, viene utilizzato solo da un'istruzione: JMP($XXXX). L'esecuzione del programma viene trasferita ad una locazione individuata sfruttando la stessa tecnica di indirezione di cui parlavamo prima. Chiarisco con un esempio: se l'istruzione in questione è JMP ($A123), il registro $A123 contiene XX ed il registro $(A123+01) contiene YY, il contatore di programma verrà caricato con il valore $YYXX, e da qui proseguirà l'esecuzione.
(Es.: JMP ($A123)=6C 23 A1)

Altre indicazioni

Conclusa la carrellata sugli indirizzamenti, scopriamo altre due colonne sulla nostra tabella che sono importantissime per la programmazione e per la comprensione del Linguaggio Macchina. Immediatamente a destra della colonna che riporta le parole chiave, viene fornita una breve descrizione, che aiuta a meglio comprendere quello che effettivamente compie l'istruzione. Tali indicazioni sono sia in inglese che in italiano, curando però di quest'ultimo la sua traduzione nella nostra lingua madre. Questa doppia nomenclatura può servire per megli abbinare mnemonicamente il nome con l'operazione eseguita. Per esempio, pensando a "BEQ", viene più spontaneo collegarlo a "Branch on EQual" che a "salta se uguale".

Oltre a ciò, vengono indicati, nell'ultima colonna a destra, tutti i singoli bits che compongono il byte di status, fondamentale per il funzionamento del calcolatore.

Nell'ordine sono riportati:
N-Negative
Z-Zero
C-Carry
I-Interrupt
D-Decimal
V-oVerflow

I bit interessati dalla determinata istruzione sono segnati con un asterisco, con uno 0 o con un 1 se lo scopo dell'istruzione è di modificarli.

LDA	M → A Carica l'accumulatore con la memoria	169	A9	2	173	AD	3	165	A5	2						
LDX	M → X Carica X con la memoria	162	A2	2	174	AE	3	166	A6	2						
LDY	M → Y Carica Y con la memoria	160	A0	2	172	AC	3	164	A4	2						
LSR	0 → [7 6 5 4 3 2 1 0] → [C]				078	4E	3	070	46	2	074	4A	1			
NOP	No OPeration Nessuna operazione														234	EA
ORA	A V M → A OR logico fra accumulatore e memoria	009	09	2	013	0D	3	005	05	2						
PHA	A → S Pone l'accumulatore nello stack														072	48
PHP	P → S Pone lo status nello stack														008	08
PLA	S → A Ritira l'accumulatore dallo stack														104	68
PLP	S → P Ritira lo status dallo stack														040	28
ROL	[C] ← [7 6 5 4 3 2 1 0] ←				046	2E	3	038	26	2	042	2A	1			
ROR	→ [7 6 5 4 3 2 1 0] → [C]				110	6E	3	102	66	2	106	6A	1			
RTI	ReTurn from Interrupt Ritorna da interrupt														064	40
RTS	ReTurn from Subroutine Ritorna da sottoprocedura														096	60
SBC	A - M - C → A , C Sottrae con carry	233	E9	2	237	ED	3	229	E5	2						
SEC	1 → C Pone a 1 il carry														056	38
SED	1 → D (esegue in decimale) Pone a 1 il modo decimale														248	F8
SEI	1 → I Pone a 1 l'interrupt														120	78
STA	A → M Pone l'accumulatore nella memoria				141	8D	3	133	85	2						
STX	X → M Pone x nella memoria				142	8E	3	134	86	2						
STY	Y → M Pone y nella memoria				140	8C	3	132	84	2						
TAX	A → X Trasferisce A in X (a non varia)														170	AA
TAY	A → Y Trasferisce A in Y (a non varia)														168	A8
TSX	S → X Trasferisce lo stack in X														186	BA
TXA	X → A Trasferisce X in A (X non varia)														138	8A
TXS	X → S Trasferisce X nello stack														154	9A
TYA	Y → A Trasferisce Y in A (Y non varia)														152	98

#	**Codice operativo decimale**	**P**	**Flag di stato**
$	**Codice operativo esadecimale**	**S**	**Stak**
N	**Numero di bytes impegnati**	**M**	**Memoria**

Attenzione alla tabella

Staccate questa doppia pagina centrale (pagina 42 e 43). Fate lo stesso con le pagine 40 e 45. Ora fate combaciare i fogli in senso orizzontale. Un po' di colla o di nastro adesivo basteranno a ricomporre la tabella, che potrete così conservare. Facile, no?

	161	A1	2	177	B1	2	181	B5	2	189	BD	3	185	B9	3										* * - - - -
													190	BE	3							182	B6	2	* * - - - -
							160	B4	2	188	BC	3													* * - - - -
							086	56	2	094	5E	3													0 * * - - -
1																									- - - - - -
	001	01	2	017	11	2	021	15	2	029	1D	3	025	19	3										* * - - - -
1																									- - - - - -
1																									- - - - - -
1																									* * - - - -
1																									DALLO STACK
							054	36	2	062	3E	3													* * * - - -
							118	76	2	126	7E	3													* * * - - -
1																									DALLO STACK
1																									- - - - - -
	225	E1	2	241	F1	2	245	F5	2	253	FD	3	249	F9	3										* * * - - *
1																									- - 1 - - -
1																									- - - - 1 -
1																									- - - 1 - -
	129	81	2	145	91	2	149	95	2	157	9D	3	153	99	3										- - - - - -
																						150	96	2	- - - - - -
							148	94	2																- - - - - -
1																									* * - - - -
1																									* * - - - -
1																									* * - - - -
1																									* * - - - -
1																									- - - - - -
1																									* * - - - -

EGENDA

∧ **AND**

∨ **OR**

⊻ **OR esclusivo**

* **Modificato**

\- **Non modificato**

M_7(M_6) Bit 7 (6) della memoria

D,X)		(IND),Y			Pag. Zero,x			Assoluto, x			Assoluto, y			Relativo			Indiretto			Pag. Zero, y			STATUS					
$	N	#	$	N	#	$	N	#	$	N	#	$	N	#	$	N	#	$	N	#	$	N	N	Z	C	I	D	V
61	2	113	71	2	117	75	2	125	7D	3	121	79	3										*	*	*	—	—	*
21	2	049	31	2	053	35	2	061	3D	3	057	39	3										*	*	—	—	—	—
					022	16	2	030	1E	3													*	*	*	—	—	—
														144	90	2							—	—	—	—	—	—
														176	BO	2							—	—	—	—	—	—
														240	FO	2							—	—	—	—	—	—
																							M_7	*	—	—	—	M_6
														048	30	2							—	—	—	—	—	—
														208	DO	2							—	—	—	—	—	—
														016	10	2							—	—	—	—	—	—
																							—	—	—	1	—	—
														080	50	2							—	—	—	—	—	—
														112	70	2							—	—	—	—	—	—
																							—	—	0	—	—	—
																							—	—	—	0	—	—
																							—	—	—	—	0	—
																							—	—	—	—	—	0
C1	2	209	D1	2	213	D5	2	221	DD	3	217	D9	3										*	*	*	—	—	—
																							*	*	*	—	—	—
																							*	*	*	—	—	—
					214	D6	2	222	DE	3													*	*	—	—	—	—
																							*	*	—	—	—	—
																							*	*	—	—	—	—
41	2	081	51	2	085	55	2	093	5D	3	089	59	3										*	*	—	—	—	—
					246	F6	2	254	FE	3													*	*	—	—	—	—
																							*	*	—	—	—	—
																							*	*	—	—	—	—
																	108	6C	3				—	—	—	—	—	—
																							—	—	—	—	—	—

SHIFT RUN PLAY
GOAL!

Si è quindi reso necessario un controllo diretto nelle locazioni dello schermo (linee 30050 e 30060).

Se il carattere risulta reserve allora viene mandato alla stampante preceduto da CHR$(18) (REVERSE ON). In caso contrario viene preceduto da CHR$(146) (REVERSE OFF). Terminato il trasferimento e chiusi i files viene reinizializzata la stampante (linea 30090). Ciò serve a riportare l'interlinea al valore di "default".

Stampa allargata o spostata

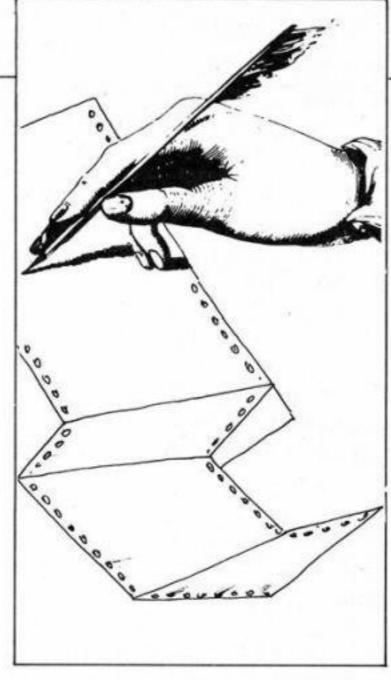
Per ottenere una copia allargata dello schermo basta inserire nella linea 30030:
PRINT 4,CHR$(14)

Tale linea risulterà quindi:
30030 FOR K=1024 TO 1984 STEP 40:
PRINT 4, CHR$(14)

E' anche possibile, entro certi limiti, fare stampare la pagina testo ad una distanza, dal margine sinistro del foglio, di vostra scelta. In tale caso la linea 30030 va cambiata così:
30030 FOR K=1024 TO 1984 STEP 40:
PRINT 4, B$

La variabile B$ deve contenere solo dei CHR$(32), cioè degli spazi. Il numero di spazi contenuti determina la distanza, dal margine sinistro, alla quale verrà stampata la schermata. Poichè una scelta della posizione tramite input sporca la pagina testo dovrete, nei vostri programmi, o prevedete tale scelta prima di disegnare la schermata oppure mantenere fisso il contenuto di B$.

In quest'ultimo caso vi basterà inserire le seguenti linee:
30020 B$="": FOR I=1 TO nn: B$=B$ + CHR$(32): NEXT
30025 IF LEN(B$) AND 1)=0 THEN B$=B$+CHR$(146)

Dove nn indica il numero di spazi (tra 1 e 40, mi raccomando! Nel caso che abbiate previsto l'input del margine, nn andrà sostituito con la variabile che contiene il valore introdotto. La linea 30025 si rende necessaria per poter stampare correttamente tutti i caratteri in reverse, nel caso che B$ contenga un numero pari a spazi.

Se siete meravigliati di tale affermazione osservate bene la stampante al lavoro, quando copia la pagina testo "spostata". Vedrete che ogni riga viene scritta in due o tre "passate". Ciò perchè la serie di caratteri comunicanti, per ogni riga di stampa, supera le 80 unità: nn spazi più 80 caratteri per lo schermo (40 di questi sono solo di controllo e quindi non fanno avanzare il carrello).

Alla fine di ogni passata viene azzerato il flag di reverse della stampante.

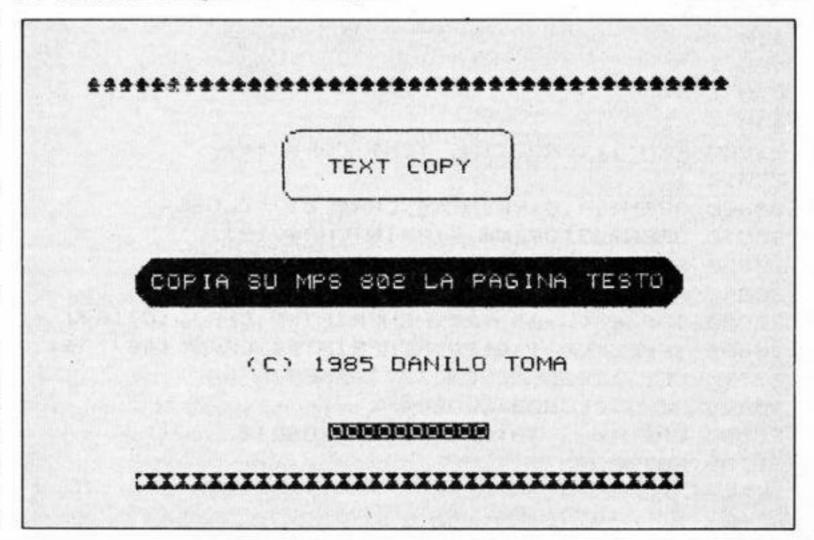
Si tratta di una specie di "carriage return with no line feed", cioè un ritorno carrello senza avanzamento della carta (vedere pag. 28 del manuale).

Ho detto "una specie" perchè il microprocessore della stampante in questo caso "sa" che la stampa va ripresa dalla colonna in cui si era interrotta e non da quella più a sinistra.

Ebbene, quando B$ contiene un numero pari di caratteri il "carriage return" capita sempre tra il carattere di controllo (REVERSE ON o REVERSE OFF) e il carattere da stampare. Quest'ultimo perciò viene sempre stampato in modo normale (REVERSE OFF). Per evitare l'inconveniente basta aggiungere un carattere a B$ quando la sua lunghezza risulti pari. Io ho usato il CHR$(146) ma va bene qualunque carattere che non sia stampabile (e che non provochi effetti indesiderati).

Rimane un problema non risolto. Se provate a riempire B$ con 39 spazi, la stampante va in crisi e non fa avanzare la carta tra una riga e l'altra. Chi è in grado di spiegare e di correggere tale anomalia?

L'ultima nota: gli apici (o virgolette) disturbano la stampa dei caratteri che li seguono, quindi cercate di evitarli nelle vostre schermate.

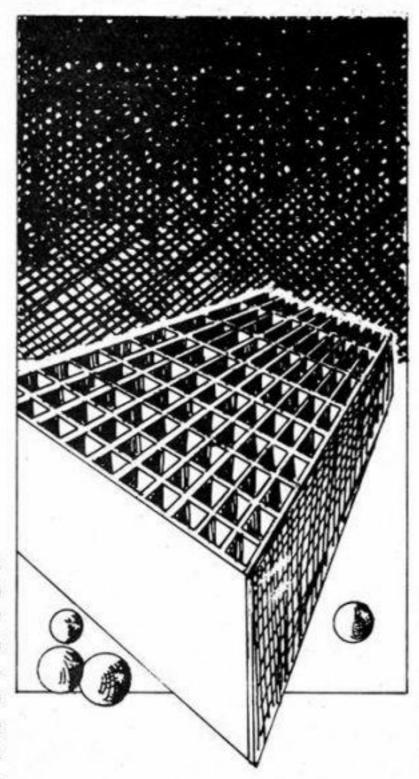
Danilo Toma

```
100 REM ******************************
110 REM **       CBM 64 & MPS 802      **
120 REM **                             **
130 REM **          TEXT COPY          **
140 REM **                             **
150 REM **   (C) 1985 DANILO TOMA      **
160 REM ******************************
170 PRINTCHR$(147);:FORI=1TO40:PRINTCHR$(97);:NEXT
180 PRINT:PRINT:PRINTTAB(12)CHR$(117);
190 FORI=1TO13:PRINTCHR$(99);:NEXT:PRINTCHR$(105)
200 PRINTTAB(12)CHR$(98);TAB(26)CHR$(125)
210 PRINTTAB(12)CHR$(98)"  TEXT COPY  "CHR$(125)
220 PRINTTAB(12)CHR$(98);TAB(26)CHR$(125)
230 PRINTTAB(12)CHR$(106);
240 FORI=1TO13:PRINTCHR$(96);:NEXT:PRINTCHR$(107)
250 PRINT:PRINT:PRINT:PRINTTAB(3)CHR$(18);CHR$(169);
260 FORI=1TO32:PRINTCHR$(32);:NEXT:PRINTCHR$(127)
270 PRINTTAB(3)CHR$(18);
280 PRINT" COPIA SU MPS 802 LA PAGINA TESTO "
290 PRINTTAB(3)CHR$(127);CHR$(18);
300 FORI=1TO32:PRINTCHR$(32);:NEXT:PRINTCHR$(146);CHR$(169)
310 PRINT:PRINT:PRINT
320 PRINTTAB(10)"(C) 1985 DANILO TOMA"
330 PRINT:PRINT:PRINT
340 PRINTTAB(15)CHR$(18);:FORI=1TO10:PRINTCHR$(119);:NEXT
350 PRINT:PRINT:PRINT
360 PRINTTAB(3)CHR$(18);:FORI=1TO34:PRINTCHR$(122);:NEXT
370 :
380 GOSUB30000: REM ** STAMPA SCHERMO **
390 :
400 END
410 :
420 :
430 :
440 :
29000 REM *** ROUTINE TEXT COPY ***
29010 :
30000 OPEN4,4,6:PRINT#4,CHR$(21):CLOSE4
30010 OPEN3,3:OPEN4,4:PRINTCHR$(19);
30030 FORK=1024TO1984STEP40
30040 FORI=0TO39:GET#3,A$
30050 IFPEEK(K+I)>127THENPRINT#4,CHR$(18);A$;
30060 IFPEEK(K+I)<128THENPRINT#4,CHR$(146);A$;
30070 NEXT:IFA$<>CHR$(13)THENPRINT#4
30080 NEXT:CLOSE3:CLOSE4
30090 OPEN10,4,10:PRINT#10:CLOSE10
30100 RETURN

READY.
```

DRIVE 1541

ARCHIVIO PER LE ROUTINE

Un metodo semplice ed efficace per utilizzare decine di routine L.M. all'interno dei vostri programmi Basic.

Supponete di scrivere un programma che utilizzi numerose routines in LM o comunque dati numerici. Un modo per scriverlo consiste nel convertire tutte le routines in numeri e trascriverle nel programma principale sotto forma di righe DATA. Questo sistema permette di avere in un corpo unico routines e programma principale, ma presenta alcuni inconvenienti.

Se, infatti, le routines sono molte e lunghe c'è un problema di spazio nella memoria RAM, che non è molto sentito in un C64, ma è abbastanza grave in un VIC 20 inespanso. Un numero inserito in una riga DATA occupa circa 2 volte la memoria che occuperebbe normalmente e, per programmi lunghi che utilizzano numerose routines, potrebbe comparire il fatidico OUT OF MEMORY, costringendoci a "restringere" il programma magari privandolo di alcune funzioni utili.

Un secondo problema è che quando si scrivono molte routines, il programma diventa un "ammasso di dati" che, oltre ad avere una scarsa leggibilità, utilizza molto tempo per leggere tutti i dati.

L'idea che proponiamo è quella di registrare tutte le routines su disco (o eventualmente su cassetta) come files sequenziali e richiamarle solo quando servono. Con questo sistema è possibile avere una routine per volta in memoria, con un notevole risparmio di locazioni RAM. Un secondo vantaggio è che le varie routines possono essere allocate anche tutte nella stessa parte di memoria, dato che vengono caricate una alla volta.

I file sequenziali

Per chi non sapesse cosa sono o come si gestiscono i file sequenziali, ecco una breve spiegazione. I file sequenziali sono un particolare modo di registrare dati numerici (e non) su disco o cassetta, l'uno dietro l'altro, appunto sequenzialmente.
Lo svantaggio consiste nel fatto che se vogliamo leggere l'ultimo dato di un file dobbiamo leggerli tutti.

Nonostante tale inconveniente, i file sequenziali possono essere utilizzati con profitto per l'applicazione descritta in questo articolo. Per cominciare a lavorare con un file sequenziale bisogna innanzitutto "aprirlo", in lettura o in scrittura con l'istruzione:

OPENnf, de, sa, "nomefile, s, r,"

... oppure...:

OPENnf, de, sa, "nomefile, s, w".

"Nf" è il numero del file che può variare a nostro piacere tra 1 e 255.

"De" è il numero della periferica sulla quale vogliamo aprire il file (1 per la cassetta, 8 per il disco).

"Sa" è l'indirizzo secondario che può essere 0 oppure 1 rispettivamente per la lettura e per la scrittura.

"Nome file" è appunto il nome del file

che vogliamo aprire, che per la cassetta è opzionale ma per il disco è obbligatorio e non deve essere più lungo di 16 caratteri, pena il troncamento degli X-16 caratteri più a destra.

",s" serve per dire che il file aperto è di tipo sequenziale.

",r" e ",w" servono per indicare se il file è in lettura (Read) o in scrittura (Write).

Esiste un terzo modo di aprire un file sequenziale in scrittura, non citato nel manuale del drive 1541. Digitate al posto di ",r" o ",w", il codice ",a" (Append). Se si apre un file in questo modo e con lo stesso nome di un file già presente su disco, i dati scritti nel file appena aperto andranno ad aggiungersi ai dati del file su disco permettendo così di concatenare due o più files.

Dopo aver aperto il file si può, a seconda dei casi, leggere o scrivere dati con le istruzioni PRINT#nf, INPUT#nf, oppure GET#nf. Esaminiamole in dettaglio:

- PRINT#nf (dove nf è il numero del file aperto) serve per scrivere un dato. La sua sintassi è:
- PRINT#nf, dato (,dato2, dato3, ecc.), in cui "dato" può essere una variabile numerica o stringa, un numero o una stringa racchiusa tra apici e chi più ne ha più ne metta. Se scriviamo con la stessa PRINT# più dati, questi devono essere separati da virgole.
- INPUT# serve per leggere un dato dal file.

La sintassi è:

INPUT#nf, var (,var2, var3, ecc.)

in cui "var" è una variabile a piacere tra quelle possibili.

- GET#serve anch'esso per prelevare dati ma, a differenza di • INPUT#, preleva un carattere alla volta.

Ad esempio se con INPUT# leggiamo il valore 257, con GET# leggeremo dapprima 2, poi 5 ed infine 7, in tre passaggi diversi. La sintassi di GET# è la stessa di INPUT#.

Infine il file DEVE essere chiuso con l'istruzione CLOSEnf, in cui "nf" è sempre il numero del file aperto.

Se si tenta di scrivere in un file aperto in lettura si avrà un NOT OUTPUT FILE ERROR, mentre se si tenta di leggere in un file aperto in scrittura si avrà il corrispondente NOT INPUT FILE ERROR.

I listati 1, 2 e 3 serviranno per chiarire meglio le idee.

Come utilizzare il programma

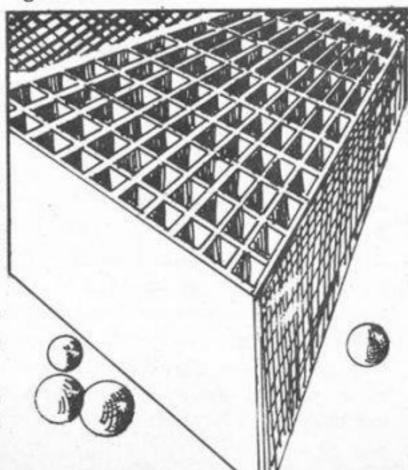
Dopo questa parentesi ritorniamo alla descrizione del programma. Per poter utilizzare le routines in questo modo bisogna innanzitutto... farle, poi trasformarle in numeri decimali ed inserirle come righe DATA in coda al programma del listato 4.

Bisogna quindi dare RUN al programma che, una volta avviato, chiederà prima l'indirizzo di partenza della routine, l'indirizzo di fine della stessa, il SYS che bisogna dare per avviarla (che non sempre coincide con l'indirizzo di partenza) e infine il nome che vogliamo dare alla routine, che sarà quello con cui dovremo poi richiamarla.

La routine verrà così scritta sul disco apposito (che avremo naturalmente provveduto ad inserire) e rimarrà a nostra disposizione per tutte le volte che ne avremo bisogno.

Per richiamarla ed eseguirla in un programma principale si deve inserire in questo la linea:

NM$="nome routine": GOSUB 63000: SYS SY

Da 63000 in poi dovranno essere inserite nel programma principale le corrispondenti linee del listato 5. Se non vogliamo eseguire immediatamente la routine basterà memorizzare SY in una variabile, ad esempio:

NM$="nome routine": GOSUB 63000: R1=SY

e digitare poi SYS R1 in qualsiasi punto del programma che utilizzi quella routine.

Questo sistema permette anche di migliorare la leggibilità del programma. Infatti leggendo il nome della routine in NM$="nome" si dovrebbe capire a che cosa serve la stessa e, di conseguenza, comprendere meglio che cosa fa quel determinato pezzo di programma.

I listati 6, 7 e 8 sono degli esempi di routines da provare, usando il programma del listato 4. I1 è l'indirizzo di partenza, I2 quello di fine e SY è la SYS da dare per avviarle. Il nome da dare a queste routines deve essere quello specificato fra apici nella seconda linea REM dei tre listati. Le righe da 5 a 300 del listato 5 servono come dimostrazione di quanto detto: caricano ed eseguono le tre routines dei listati 6, 7, 8 seguendo il metodo prima descritto. Tali routines possono apparire di scarsa utilità ed infatti hanno il solo scopo di realizzare un semplice DEMO di applicazione.

Il numero massimo di routines registrabili su un disco è 144, perché questo è appunto il numero di files che la directory di un disco formattato con il 1541 può contenere.

Nella directory le nostre routines avranno il suffisso SEQ (SEQuential= sequenziale) al posto del solito PRG (PRoGram) che siamo abituati a vedere. Le routines possono anche essere registrate su cassetta semplicemente aprendo un file con de=1, invece che con de=8, ma il tempo di lettura dei files sequenziali da cassetta è talmente elevato da rendere molto più conveniente il "vecchio" sistema delle linee DATA.

I più esperti potranno modificare il programma in modo da usarlo per registrare dati che non siano routines LM (ad es. dati per sprite, per la musica, ecc.) battendo RETURN a vuoto alla domanda 'SYS' e ovviamente non facendo SYS SY in fase di ricarica dei dati stessi.

Giancarlo Mariani

ARCHIVIO ROUTINE 1

```
10 REM  "LISTATO 1"
15 REM  "SCRITTURA DI UN FILE SEQ
   UENZIALE"
20 PRINT"INSERISCI 5 PAROLE"
30 FOR  K=1  TO  5:INPUT  A$(K):N
   EXT K
40 OPEN  1,8,1,"PAROLE,S,W":REM
   APERTURA DEL FILE SU DISCO
50 FOR  K=1  TO  5
60 PRINT#1,A$(K):REM  SCRITTURA P
   AROLE
70 NEXT K
80 CLOSE  1:REM  CHIUSURA DEL FIL
   E
```

ARCHIVIO ROUTINE 2

```
10 REM  "LISTATO 2"
15 REM  "LETTURA DI UN FILE CON IN
   PUT#"
20 OPEN 1,8,0,"PAROLE,S,R":REM  AP
   RE IL FILE IN LETTURA
30 FOR  K=1  TO  5
40 INPUT#1,A$(K):REM  CARICA LE PA
   ROLE
50 NEXT K
60 CLOSE  1:REM  CHIUDE IL FILE
70 FOR  K=1  TO  5
80 PRINT A$(K):REM  SCRIVE LE PARO
   LE SUL VIDEO
90 NEXT K
```

ARCHIVIO ROUTINE 3

```
10 REM  "LISTATO 3"
20 REM  "LETTURA DI UN FILE CON GE
   T#"
25 OPEN 1,8,0,"0:PAROLE,S,R":REM
   COME LISTATO 2
30 GET #1,A$: REM  CARICA UNA LETT
   ERA ALLA VOLTA...
40 PRINT A$;:REM  ...E LA SCRIVE S
   U VIDEO
50 REM  CONTROLLO,TRAMITE LA VARI
   ABILE DI SISTEMA ST,CHE SI SIA
   LETTO L'ULTIMO
60 REM  L'ULTIMO DATO DEL FILE. (
   VALORE NORMALE ST=0)
70 IF  ST=0  THEN  30:REM  SE ST=0
    VA A 30,SE NO FINISCE
80 CLOSE  1:REM  CHIUDE IL FILE
```

ARCHIVIO ROUTINE 4

```
1 REM  SCRITTURA ROUTINE SU DISCO
2 REM  BY MARIANI GIANCARLO
3 :
5 PRINT"[CLEAR]"
7 INPUT "INDIRIZZO PARTENZA ";I1
9 INPUT "INDIRIZZO FINE     ";I2:
  IF I2<I1 THEN 9
11 INPUT "SYS               ";SY
13 INPUT "NOME ROUTINE      ";N$:
   IF LEN(N$)>16 THEN 13
19 OPEN 2,8,15
20 OPEN 1,8,1,"0:"+N$+",S,W"
30 PRINT#1,I1:PRINT#1,I2:PRINT#1,S
   Y
40 FOR  K=I1  TO  I2:READ  A:PRINT
   #1,A:NEXT
50 INPUT#2,A$,S$,D$,F$:PRINTA$ S$
   D$ F$
60 CLOSE 1:CLOSE 2
65 :
70 REM  INSERIRE IN QUESTE RIGHE
80 REM  LE ISTRUZIONI "DATA" DEL
90 REM  PROGRAMMA IN L.M.
100 REM  (VEDI ESEMPI PUBBLICATI)
```

ARCHIVIO ROUTINE 5

```
 5 REM  LISTATO 5
10 REM  LETTURA+DIMOSTRAZIONE
15 REM  BY MARIANI G.
20 PRINT"[CLEAR]"
30 NM$="INTESTAZIONE":GOSUB 63000:
   SYS SY
40 GOSUB  300
50 NM$="ARCOBALENO":GOSUB  63000:S
   YS SY
60 GOSUB  300
70 NM$="COLORI":GOSUB  63000:SYS S
   Y
80 GOSUB  300
90 PRINT"[CLEAR][DOWN]FINE DIMOSTR
   AZIONE":END
```

```
  300 POKE 198,0:WAIT 198,1:RETURN
62990 END
62995 REM   "LETTURA ROUTINE LM."
62996 REM   "IL NOME DEVE ESSERE IN NM
      $."
62997 REM   "DOPO AVERLA CARICATA,"
62998 REM   "LA ROUTINE SI PUO' ESEGUI
      RE"
62999 REM   "DIGITANDO SYS SY."
63000 :
63005 OPEN 71,8,0,"0:"+NM$+",S,R"
63010 INPUT#71,I1:INPUT#71,I2:INPUT#7
      1,SY
63020 FOR K=I1 TO I2:INPUT#71,A:POKE
      K,A:NEXT:CLOSE 71:RETURN
```

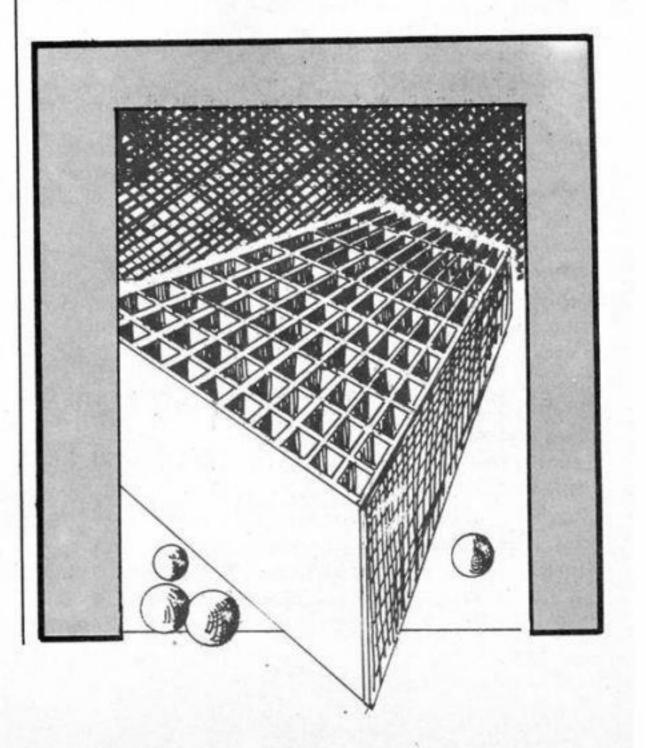
ARCHIVIO ROUTINE 6

```
 985 REM  LISTATO 6
 990 REM  "INTESTAZIONE"
 992 REM  I1=49152:I2=49332:SY=49152
1000 DATA 169,0,170,168,189,21,192,1
     57
1010 DATA 40,4,169,14,157,40,216,232
1020 DATA 224,160,208,240,96,32,32,3
     2
1080 DATA 32,32,32,32,32,32,32,1
1090 DATA 18,3,8,9,22,9,15,32
1100 DATA 19,21,2,18,15,21,20,9
1110 DATA 14,5,32,32,32,32,32,32
1120 DATA 32,32,32,32,32,32,32,32
1130 DATA 32,32,32,32,32,32,2,25
1140 DATA 32,32,13,1,18,9,1,14
1150 DATA 9,32,7,9,1,14,3,1
1160 DATA 18,12,15,32,32,32,32,32
1170 DATA 32,32,32,32,32,32,32,32
1180 DATA 22,9,1,12,5,32,2,18
1190 DATA 9,1,14,26,1,32,55,50
1200 DATA 32,32,50,48,48,51,54,32
1210 DATA 13,5,4,1,32,40,13,9
1220 DATA 41,32,32,32,32,32,32,32
1230 DATA 32,32,32,32,32,32,32,32
1240 DATA 32,20,5,12,46,32,48,51
1250 DATA 54,50,47,55,50,53,54,53
1260 DATA 32,32,32,32,32,32,32,32
1270 DATA 32,32,32,32,32
```

ARCHIVIO ROUTINE 7

```
 850 REM  LISTATO 7
 900 REM  "ARCOBALENO"
 910 REM  "I1=49152:I2=49197:SY=4915
     2"
 920 REM
1000 DATA 169,42,162,0,160,0,157,0
1010 DATA 4,157,250,4,157,244,5,157
1020 DATA 238,6,152,157,0,216,157,25
     0
1030 DATA 216,157,244,217,157,238,21
     8,200
1040 DATA 192,16,208,2,160,0,169,42
1050 DATA 232,224,250,208,217,96
```

ARCHIVIO ROUTINE 8

```
 850 REM  LISTATO 8
 900 REM  "COLORI"
 910 REM  "I1=49152:I2=49179:SY=4915
     2"
 920 :
1000 DATA 173,32,208,133,251,162,0,1
     60
1010 DATA 0,142,32,208,232,224,255,2
     08
1020 DATA 248,200,192,255,208,243,16
     5,251
1030 DATA 141,32,208,96
```

GESTIONE DEI NOMINATIVI

Ordinamento alfabetico, editing, registrazione e stampa di un elenco di nomi.

Nel descrivere l'utilizzo del programma pubblicato si seguirà un ordine totalmente diverso da quello che compare nel menù. Privilegiando, cioè, l'ordine "logico".

- Opzione 2. Aggiunge nomi alla lista, se presente, altrimenti ne crea una. Si devono introdurre i nomi seguiti da RETURN mentre si ritorna al menù digitando l'asterisco (*).
- Opzione 7. Mostra la lista dei nomi presenti in memoria. Con un tasto qualsiasi si passa alla pagina successiva, mentre con R si ritorna al menù principale.
- Opzione 3. Si usa per cancellare nomi scritti in modo errato o che non si vogliono più considerare. Quando compare il nome, premendo il tasto "S" verrà confermato, con "N" cancellato mentre con "R" verrà eseguita l'operazione di cancellatura con successivo ritorno al menù.
- Opzione 1. Ordina alfabeticamente la lista di nomi presenti in memoria.
- Opzione 4. Registra i nomi su supporto magnetico ed è seguita da un menù secondario che richiederà se i nomi si vogliono aggiungere (merge) o sostituire (replace) ad un eventuale file già presente (se si vuole creare un nuovo file usare replace). In seguito verrà chiesto il nome del file. Con l'opzione merge, il nuovo file verrà caricato automaticamente in memoria.
- Opzione 5. Carica in memoria un file in cui verrà richiesto il nome.
- Opzione 6. Stampa la lista dei nomi presente in memoria.

Verranno poste alcune domande: se alla prima di esse si risponde con "C", verrà eseguita una stampa compattata incolonnata su tre colonne e numerata.
Rispondendo invece con "N", verrà posta un'altra domanda.
Digitando "N", si avrà una stampa non numerata su un numero di colonne a scelta il cui numero massimo, calcolato automaticamente in base alla lunghezza massima dei nomi, sarà comunicato volta per volta, mentre digitando "2" si avrà una stampa numerata su 2 colonne.

Facciamo notare che l'opzione di stampa compattata è utilizzabile solo in unione con la stampante MPS-801. Disponendo di altre stampanti (come la MPS-802) è possibile solo l'opzione di stampa normale.

- Opzione 8. Fine del programma.
- Opzione 9. Riporta il programma allo stato di partenza.

Si possono impostare fino a 1000 nomi ma in tal caso è consigliabile compilare il programma.

A proposito di Mps 801...

Nel programma viene utilizzata la subroutine di stampa compattata pubblicata sul N. 15 di Commodore Computer Club.
Lo sapete proprio tutti come si disegna con la MPS 801?
Ai principianti viene fornita qui di seguito una breve spiegazione riguardo questo argomento.

Per programmare un carattere non standard come quello di figura 1, bisogna farne una raffigurazione su di un foglio a quadretti utilizzando una striscia alta 7 quadretti e lunga fino a 480 (!!). Si anneriscono i quadretti in cui si desidera inserire un punto mentre si lasciano bianchi i rimanenti. Finito il disegno inizia la parte dedicata al calcolo.

Si considera una colonna per volta (striscia alta 7 e lunga 1 quadretto) e a partire dall'alto si moltiplica il contenuto di ogni quadretto (pieno=1 vuoto=0) per le potenze crescenti di 2 (da 2 elevato 0 fino a 2 elevato 6). In seguito si sommano tutti i valori della colonna ottenuti, con il numero costante 128 e si prende nota del risultato. Nella figura 1 il procedimento è rappresentato, per motivi di spazio, solo per le prime

due colonne. Per le altre il procedimento usato è perfettamente identico.

Per stampare il carattere programmato, si apre dapprima un canale (comando OPEN) e si invia il carattere che segnala l'ingresso in graphic mode: CHR$(8), seguito da un punto e virgola (;). Si inviano quindi uno alla volta i dati in forma ASCII (il solito CHR$(X)) seguiti ciascuno dal punto e virgola. Alla fine si invia un CHR$(15) (col punto e virgola se si vuole stampare di seguito) per tornare al modo di stampa normale.

Volendo seguire un disegno che si estenda in verticale per più di 7 punti è sufficiente dividere il disegno nelle solite strisce di 7 quadretti, mentre per segnalarne la fine (cioè per andare a capo) è necessario inviare il dato di valore 13 (13, NON 13+128).

Fabio Sorgato

```
1 .  ..●●....●●.....●.
2    .●..●●.●..●...●●.
4    ●...●..●.....●.●.
8    ●......●●●..●..●.
16   ●...●..●..●.●..●.
32   .●..●●.●..●.●●●●●
64   ..●●....●●.....●.
128  _________________
     1234

1:1*0+2*0+4*1+8*1+16*1+32*0+64*0+128=156

2:1*0+2*1+4*0+8*0+16*0+32*1+64*0+128=162

10 OPEN1,4:PRINT#1,CHR$(8);
20 FORK=1TO17:READA:PRINT#1,CHR$(A);:NEXT
30 PRINT#1,CHR$(15);"THE NEW COMPUTER":
   CLOSE1:END
40 DATA156,162,193,193,182,162,128,190,201
   ,201,178,128,184,164,162,255,160

RUN

C64 THE NEW COMPUTER
```

```
------------------------------------------------------------------------------------
|ALESSANDRO|FABRIZIO  |GENNARO   |MARIANNA  |PAOLA     |          |
|BETTINO   |FRANCESCO |LIVIA     |MASSIMO   |SANDRO    |          |
|ERMANNO   |GASPARE   |MARIA     |MIRELLA   |SILVANA   |          |
------------------------------------------------------------------------------------
------------------------------------------------------------------------------------
| 1) ALESSANDRO                | 6) GASPARE                   | 11) MASSIMO
| 2) BETTINO                   | 7) GENNARO                   | 12) MIRELLA
| 3) ERMANNO                   | 8) LIVIA                     | 13) PAOLA
| 4) FABRIZIO                  | 9) MARIA                     | 14) SANDRO
| 5) FRANCESCO                 |10) MARIANNA                  | 15) SILVANA
------------------------------------------------------------------------------------
```

```
10 REM  **          GESTIONE    NOMI
          **
20 REM  ** COMPUTER COMMODORE & M
   PS 801 **
30 REM  **
          **
40 :
100 PRINT"[CLEAR]":DIM  PR$(1000):
   GOSUB 1780
110 N=0:A$="":K1=0:K=0:C=0:OP$="":
   C1=0
115 C3=0:K2=0:K3=0:C2=0:M=0:M1=0:B
   $="":C$=""
120 D$="":DC=0:BO=0:F=0:DI$="":RR$
   =""
125 QW$=CHR$(8)+CHR$(128)+CHR$(255
   )+CHR$(128)
130 WW$="":KK$="":J=0
140 PRINT"[CLEAR][2 DOWN]";
150 PRINT"1)-ORDINA ALFABETICAMENT
   E"
160 PRINT"2)-AGGIUNGE NOMI"
170 PRINT"3)-TOGLIE NOMI"
180 PRINT"4)-REGISTRA NOMI"
```

```
190 PRINT"5)-LEGGE NOMI"
200 PRINT"6)-STAMPA NOMI"
210 PRINT"7)-MOSTRA NOMI"
220 PRINT"8)-FINE"
230 PRINT"9)-RESET"
240 PRINT"[3 DOWN]":POKE 198,0
250 INPUT "QUALE (1-9)";A$:IF A$<"
    1" OR A$>"9" THEN A$="":PRINT"
    [2 UP]":GOTO 250
260 IF A$="9" THEN 280
270 GOTO 320
280 PRINT"[CLEAR][2 DOWN]SEI SICUR
    O? (S/N)":POKE 198,0
290 GET A$:IF A$<>"S" AND A$<>"N"
    THEN 290
300 IF A$="S" THEN  RUN
310 GOTO 140
320 ON VAL(A$)GOTO 330,480,560,730
    ,940,1040,1590,1720
330 REM  ***ORDINA  ALFABETICAMENT
    E***
340 IF N<=1 THEN 140
350 PRINT"[CLEAR][DOWN]ORDINAMENTO
     ALFABETICO"
360 PRINT"[DOWN]ATTENDERE PREGO...
    [DOWN]"
370 FOR C1=N TO 2 STEP -1
380 OP$=PR$(1)
390 C3=1
400 FOR C2=2 TO C1
410 IF PR$(C2)>OP$ THEN OP$=PR$(C2
    ):C3=C2
420 NEXT
430 PR$(C3)=PR$(C1)
440 PR$(C1)=OP$
450 NEXT
460 PRINT"[DOWN]FATTO!"
470 PRINT"[DOWN]PREMI UN TASTO[DOW
    N]":POKE 198,0:WAIT 198,1:GOTO
     140
480 REM  ***  AGGIUNGE  NOMI  ***
490 N=N+1:PRINT"[CLEAR][DOWN]AGGIU
    NGE NOMI[DOWN]"
500 PRINT"(* PER FINIRE)[2 DOWN]"
510 REM
520 PR$(N)="":PRINT"NOME N.";N;:IN
    PUT PR$(N)
530 IF PR$(N)="*" THEN PR$(N)="":N
    =N-1:GOTO 140
540 IF PR$(N)="" THEN PRINT"[2 UP]
    ":GOTO 510
550 N=N+1:GOTO 510
560 REM  ** TOGLIE NOMI **
570 PRINT"[CLEAR][DOWN]TOGLIE NOMI
    [DOWN]"
580 K=1:C=1:PRINT"[CLEAR][DOWN]([R
    VS]S[RVOFF]I,[RVS]N[RVOFF]O,[R
    VS]R[RVOFF]ITORNO)[2 DOWN]"
590 PRINT"NOME N."K":";PR$(K)
600 GET A$:IF A$<>"S" AND A$<>"N"
    AND A$<>"R" THEN 600
610 IF A$="S" THEN PRINT TAB(37);"
    [UP]SI"
620 IF A$="R" THEN 650
630 IF A$="N" THEN C=C+1:GOTO 720
640 K=K+1:IF K<=N THEN 590
650 PR$(N+1)="[PI]":FOR K=1 TO N
660 IF PR$(K)="" THEN 690
670 NEXTK:N=N-C+1:IF N<=1 THEN N=1
680 GOTO 710
690 FOR K1=K TO N:PR$(K1)=PR$(K1+1
    )
700 NEXTK1:GOTO 660
710 PRINT"[DOWN]PREMI UN TASTO":PO
    KE 198,0:WAIT 198,1:GOTO 140
720 PRINT TAB(37)"[UP][RVS]NO":PR$
    (K)="":GOTO 640
730 REM  ***  REGISTRA  NOMI ***
740 IF N<=0 THEN PRINT"[CLEAR][2 D
    OWN]NON CI SONO NOMI IN MEMORI
    A!![DOWN]":GOTO 1710
750 PRINT"[CLEAR][DOWN]REGISTRAZIO
    NE NOMI[DOWN]"
760 PRINT"[DOWN][RVS]M[RVOFF]ERGE
    O [RVS]R[RVOFF]EPLACE[DOWN]"
770 GET A$:IF A$<>"M" AND A$<>"R"
    THEN 770
780 IF A$="M" THEN 870
790 INPUT "NOME FILE";A$
800 PRINT"ATTENDERE PREGO..."
810 OPEN 1,8,15,"S0:"+A$:CLOSE 1
820 OPEN 1,8,1,""+A$:PRINT#1,N
830 FOR K=1 TO N+1:PRINT#1,PR$(K)
840 NEXT:PRINT#1,CHR$(13):CLOSE 1
850 PRINT"[2 DOWN]FATTO!"
860 PRINT"[DOWN]PREMI UN TASTO[DOW
    N]":POKE 198,0:WAIT 198,1:GOTO
     140
870 REM  MERGE
880 INPUT "NOME FILE";A$
890 PRINT"ATTENDERE PREGO..."
900 OPEN 1,8,0,A$
```

```
910 INPUT#1,C
920 FOR K=1+N TO C+N+1:INPUT#1,PR$
    (K):NEXT:CLOSE 1:N=N+C
930 GOTO 810
940 REM  ***    LEGGE NOMI     ***
950 PRINT"[CLEAR][DOWN]LETTURA NOM
    I[DOWN]"
960 INPUT "NOME FILE";A$
970 PRINT"ATTENDERE PREGO...":OPEN
     1,8,15,"I0":CLOSE 1
980 OPEN 1,8,0,A$
990 INPUT#1,N
1000 FOR K=1 TO N:INPUT#1,PR$(K)
1010 NEXT:CLOSE 1
1020 PRINT"[2 DOWN]FATTO!"
1030 PRINT"[DOWN]PREMI UN TASTO[DOW
     N]":POKE 198,0:WAIT 198,1:GOTO
      140
1040 REM  ***   STAMPA  NOMI   ***
1050 PRINT"[CLEAR][DOWN]STAMPA NOMI
     "
1060 INPUT "[RVS]N[RVOFF]ORMALE O [
     RVS]C[RVOFF]OMPATTA";W$:IF W$<
     >"N" AND W$<>"C" THEN 1040
1070 IF W$="C" THEN 1350
1080 INPUT "[CLEAR][RVS]2[RVOFF] FI
     LE NUMERATA O [RVS]N[RVOFF] FI
     LE";W$:IF W$<>"N" AND W$<>"2"
     THEN 1080
1090 IF W$="N" THEN 1220
1100 PRINT"[DOWN]ATTENDERE PREGO...
     [2 DOWN]":PR$(N+1)=""
1110 M=INT(N/2):M1=M:IF N/2>M THEN
     M1=M1+1
1120 OPEN 1,4
1130 FOR K=1 TO 80:PRINT#1,"-";:NEX
     T:PRINT#1:PRINT#1
1140 WK$="
                         ":REM  40 SPAZI
1150 FOR K=1 TO M1
1160 A$=LEFT$("| "+STR$(K)+") "+PR$(
     K)+WK$,39)
1170 D$=A$
1180 A$=LEFT$("| "+STR$(K+M1)+") "+P
     R$(K+M1)+WK$,39)
1190 PRINT#1,D$;A$:NEXT:PRINT#1
1200 FOR K=1 TO 80:PRINT#1,"-";:NEX
     T:PRINT#1:CLOSE 1
1210 PRINT"[DOWN]PREMI UN TASTO[DOW
     N]":POKE 198,0:WAIT 198,1:GOTO
      140
1220 FOR K=1 TO N:MA=LEN(PR$(K)):IF
      MA>MF  THEN  MF=MA
1230 NEXT:MF=INT(80/(MF+1))
1240 PRINT"[CLEAR]QUANTE FILE (MAX"
     ;MF;")";:INPUT  FI:IF  FI<1  O
     R  FI>MF  THEN 1240
1250 PRINT"[DOWN]ATTENDERE PREGO...
     [2 DOWN]"
1260 M1=INT(N/FI):IF M1*FI<>N THEN
     M1=M1+1
1270 OPEN 1,4:SI=INT(80/FI):FOR K=N
     +1 TO N+FI+1:PR$(K)="":NEXT
1280 FOR K=1 TO SI*FI:PRINT#1,"-";:
     NEXT:PRINT#1:PRINT#1
1290 WK$="
                         ":REM  40 SPAZI
1300 FOR K=1 TO M1:FOR J=0 TO FI-1
1310 A$=LEFT$("|"+PR$(K+J*M1)+WK$,S
     I)
1320 PRINT#1,A$;:NEXT:PRINT#1:NEXT:
     PRINT#1
1330 FOR K=1 TO SI*FI:PRINT#1,"-";:
     NEXT:PRINT#1:CLOSE 1
1340 PRINT"[DOWN]PREMI UN TASTO[DOW
     N]":POKE 198,0:WAIT 198,1:GOTO
      140
1350 REM  *** STAMPA COMPATTA ***
1360 PRINT"[DOWN]ATTENDERE PREGO...
     [2 DOWN]":PR$(N+1)=""
1370 PR$(N+2)=""
1380 OPEN 1,4:PRINT#1,CHR$(15);
1390 FOR K=1 TO 77:PRINT#1,"-";:NEX
     T:PRINT#1
1400 WW$="
                 "
1410 FOR H=1 TO 3:PRINT#1,QW$+CHR$(
     128)+CHR$(128);:GOSUB 1520:NEX
     T
1420 PRINT#1,QW$+CHR$(13);
1425 REM  WK$=52 SPAZI
1430 WK$="
                                 ":R
     EM
1440 M1=INT(N/3):IF M1*3<>N THEN M1
```

```
    =M1+1
1450 FOR L=1 TO M1:PRINT#1,QW$;:FOR
      K=0 TO 2:J=L+M1*K
1460 WW$=STR$(J)+") ":WW$=WW$+LEFT$
     (PR$(J)+WK$,50)
1470 GOSUB 1550:WW$=LEFT$(KK$,150):
     GOSUB 1570
1480 WW$=QW$:GOSUB 1570
1490 NEXT:PRINT#1,CHR$(13);:NEXT
1500 PRINT#1,CHR$(15);:FOR K=1 TO 7
     7:PRINT#1,"-";:NEXT:PRINT#1:CL
     OSE 1
1510 PRINT"[DOWN]PREMI UN TASTO[DOW
     N]":POKE 198,0:WAIT 198,1:GOTO
      140
1520 REM  *** COMPATTA LA STRINGA *
     **
1530 PRINT#1,CHR$(8);:FOR T=1 TO LE
     N(WW$)
1535 PRINT#1,A$(ASC(MID$(WW$,T,1)))
     ;:NEXT
1540 RETURN
1550 REM
1560 KK$="":FOR T=1 TO LEN(WW$):KK$
     =KK$+A$(ASC(MID$(WW$,T,1))):NE
     XT:RETURN
1570 REM  ** STAMPA COMPATTA **
1580 PRINT#1,CHR$(8)+WW$;:KK$="":WW
     $="":RETURN
1590 REM  *** MOSTRA NOMI ***
1600 PRINT"[CLEAR][DOWN]MOSTRA NOMI
     [DOWN]":C=0
1610 PRINT"[DOWN]PREMI UN TASTO":PO
     KE 198,0:WAIT 198,1:POKE 198,0
1620 PRINT"[CLEAR]      NOME[DOWN]"
1630 FOR K=1 TO N:PRINTK") "PR$(K)
1640 C=C+1:IF C>=20 THEN 1660
1650 NEXT:GOTO 1710
1660 C=0:PRINT"[DOWN]PREMI UN TASTO
      ([RVS]R[RVOFF] PER FINIRE)"
1670 POKE 198,0
1680 GET A$:IF A$="" THEN 1680
1690 IF A$="R" THEN 140
1700 PRINT"[CLEAR]      NOME[DOWN]":
     GOTO 1650
1710 PRINT"[DOWN]PREMI UN TASTO":PO
     KE 198,0:WAIT 198,1:POKE 198,0
     :GOTO 140
1720 REM  ***           FINE
      ***
1730 PRINT"[CLEAR][2 DOWN]SEI SICUR
     O? (S/N)":POKE 198,0
1740 INPUT A$:IF A$="N" THEN 140
1750 IF A$="S" THEN PRINT"[2 DOWN]C
     IAO!!":CLR :END
1760 GOTO 1730
1770 REM  * DEFINISCE CARATTERI COM
     PATTATI *
1780 DIM A$(128):T=32
1790 READ A:IF A<128 THEN J=128
1800 A$(T)=A$(T)+CHR$(A+J):IF A=0 T
     HEN T=T+1
1810 J=0:IF A<0 THEN  1830
1820 GOTO  1790
1830 RETURN
1840 DATA 128,128,128,0,23,0,3,128,
     3,0,10
1845 DATA 31,10,31,10,0,18,21,31,21
     ,9,0,17,8,4,2
1850 DATA 17,0,26,21,26,20,0,2,1,0,
     14,17
1855 DATA 0,17,14,0,21,14,31,14,21,
     0,4,4,31,4,4,0
1860 DATA 16,12,0,4,4,4,4,0,16,0,24
     ,4,3,0
1865 DATA 31,17,31,0,31,0,29,21,23,
     0,17,21,31,0
1870 DATA 7,4,31,0,23,21,13,0,31,21
     ,29,0,1,1,31,0,31,21,31,0,23,2
     1,31,0
1880 DATA 18,0,16,10,0,4,10,17,17,0
     ,10,10,10,10,0,17,17,10,4,0,2,
     1,21,5,2,0
1890 DATA 31,17,21,23,0,31,5,31,0,1
     7,31,21,31,0,31,17,17,0,31,17,
     27,14,0
1900 DATA 31,21,17,0,31,5,1,0,31,17
     ,29,0
1905 DATA 31,4,31,0,17,31,17,0,17,3
     1,1,0,31,6,25
1910 DATA 0,31,16,16,0,31,2,4,2,31,
     0
1920 DATA 31,2,4,31,0,31,17,31,0,31
     ,5,7,0
1925 DATA 31,17,31,16,0,31,13,23,0,
     23,21,29,0
1930 DATA 1,31,1,0,31,16,31,0,15,24
     ,15,0,31,8,4,8,31,0,27,4,27,0
1940 DATA 3,30,3,0,25,21,19,0,31,17
     ,0,30,21,21,18,0,17,31,0
1950 DATA 2,31,2,0,8,28,8,8,8,0
1960 DATA -1
```

ASTA CHE CADE

Osservare la caduta di un'asta da un muro in diverse condizioni di velocità e gravità.

Il movimento di un corpo è caratterizzato da due grandezze fondamentali la velocità e l'accelerazione.

La prima è intuitivamente molto facile da capire e non esprime altro che la rapidità con cui varia la posizione di un oggetto. Lo spostamento di un oggetto, ovvero lo spazio che esso percorre, viene misurato in metri, mentre il tempo impiegato verrà, in genere, misurato in secondi.

La velocità viene quindi definita come il rapporto fra lo spazio percorso ed il tempo impiegato per percorrerlo:

v=s/t

L'accelerazione, invece, è la variazione di velocità di un determinato arco di tempo. L'accelerazione verrà quindi indicata come rapporto fra la variazione di velocità ed il tempo:

a=(V-Vo)/s

Lo spazio che il corpo percorrerà, sottoposto ad accelerazione costante, sarà:

s=t↑2*a/2

Secondo le leggi della meccanica tradizionale lo spostamento di un corpo è generato da una o più forze che agiscono su esso. La forza a cui siamo comunemente sottoposti è quella gravitazionale. Un corpo sottoposto ad una forza tenderà a spostarsi nella sua direzione.

Quindi un corpo, che nello stato di quiete viene sottoposto all'azione di una forza, subirà una accelerazione che sarà direttamente proporzionale alla forza e inversamente proporzionale alla sua massa.

F=m*a

La forza è indicata normalmente con F e si misura in newton (= intensità di forza che agendo su un corpo di massa 1Kg, gli imprime un'accelerazione di m/s↑2).

In particolare la forza di gravità si può ritenere costante e, l'accelerazione che imprime ad un corpo di massa qualsiasi, per la legge di gravitazione universale, è di circa 9.8 m/s↑2. Tale valore è chiamato costante di gravità ed è generalmente indicato con la lettera "g". Lasciando cadere un corpo in caduta libera esso sarà continuamente sottoposto ad una forza di uguale intensità.

Di conseguenza il suo sarà un moto uniformemente accelerato diretto verso il basso perpendicolarmente al terreno e viene detto "moto naturalmente accelerato".

Il programma

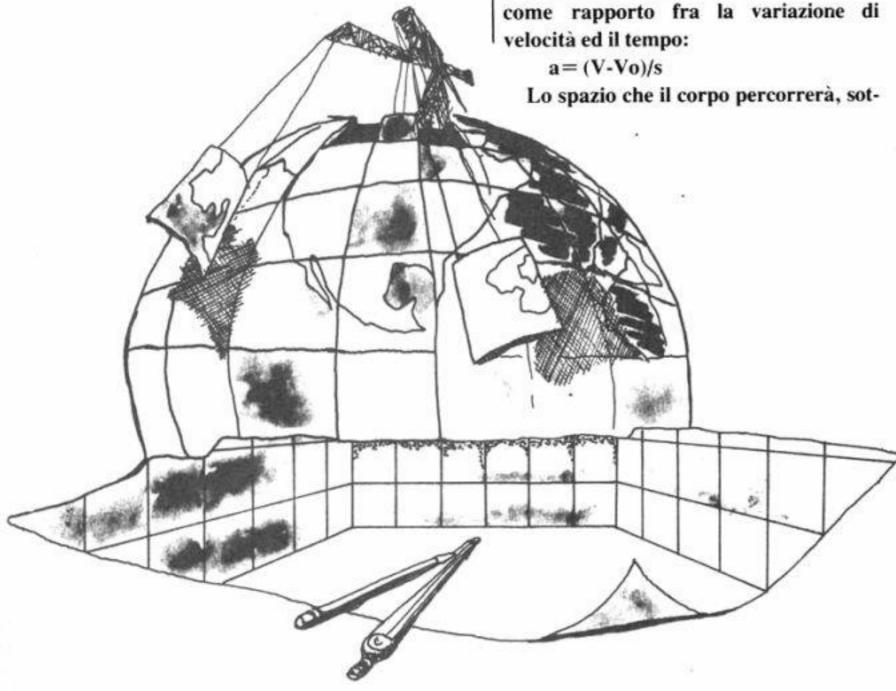
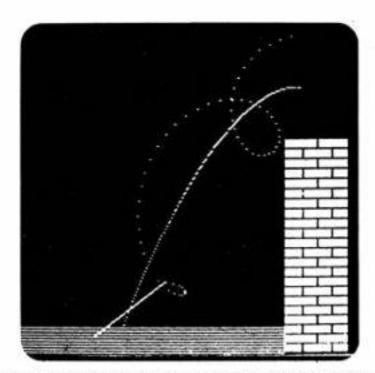
Nel nostro caso il programma simula la caduta di un'asta sottoposta all'attrazio-

Un esempio delle traiettorie che può assumere l'asta in caduta. Possiamo indifferentemente modificare la velocità di caduta e di lancio.

Figura 1: parabola dell'asta in caduta.

ne gravitazionale. Per semplificare i calcoli si suppone che la caduta avvenga in assenza di aria.

Poichè orizzontalmente il moto dell'asta è rettilineo uniforme, il suo percorso nell'intervallo di tempo "t" sarà:

x=v*t

Verticalmente, il moto è naturalmente accelerato, per cui lo spazio percorso dall'asta nello stesso intervallo di tempo "t" sarà:

y=t↑2*g/2

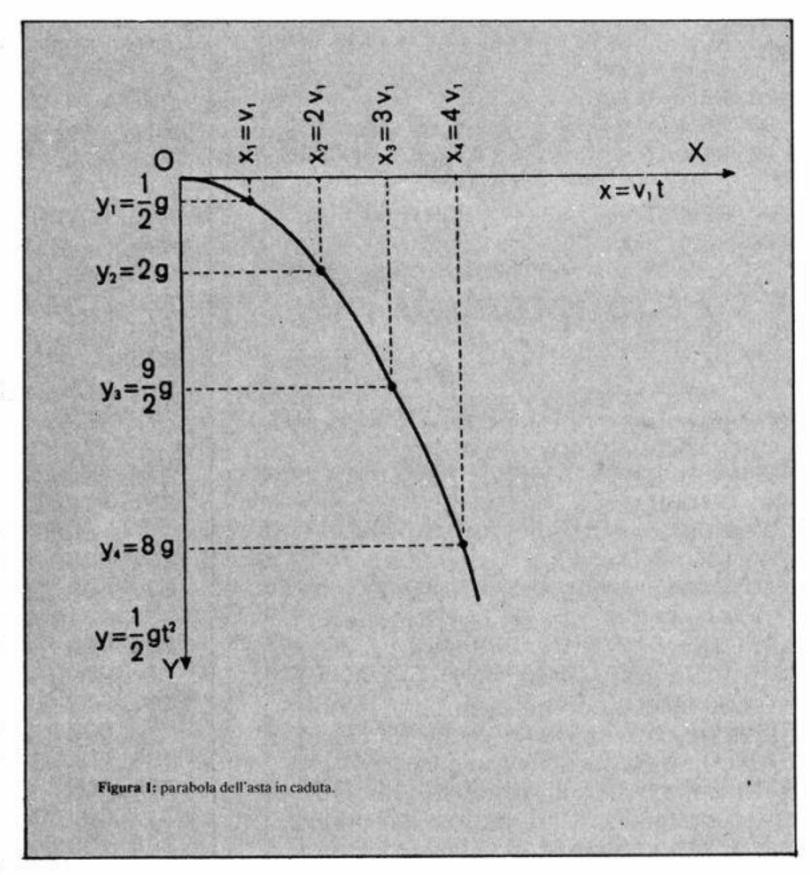
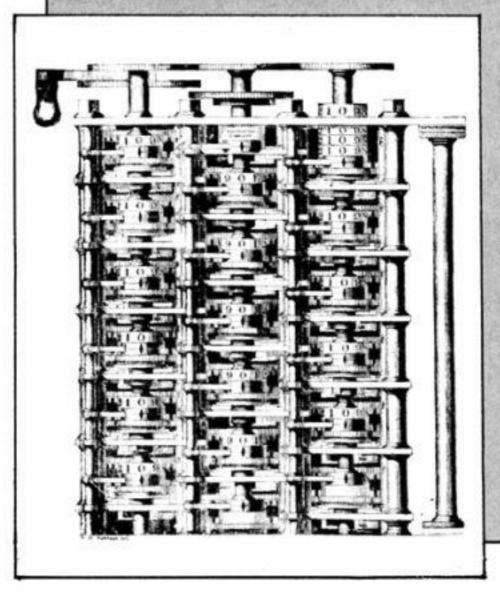
Quindi lo spostamento dell'asta avverrà, come potrete vedere, lungo una parabola (vedi figura 1). Il programma è diviso in tre parti che simulano tre diversi tipi di caduta dell'asta.

Nel primo caso essa cade verticalmente mentre nel secondo orizzontalmente.

Nel secondo, invece, essa cade compiendo una rotazione intorno al proprio centro che coincide con il centro di massa. In tutti e tre i casi si dimostra che: "il baricentro di un grave lanciato orizzontalmente percorre una traiettoria parabolica".

Nel terzo esempio è possibilie cambiare i valori dell'accelerazione (verticale) e della velocità (orizzontale).

Si può così simulare la caduta di un'asta sulla terra, sulla luna o su marte, cambiando semplicemente i valori di g.

A. & A. Boriani

LE IMMAGINI DI QUESTO FASCICOLO

Per l'inglese Charles Babbage, il problema era calcolare e stampare automaticamente le tavole nautiche ed astronomiche necessarie alla navigazione per mare. Ecco un disegno della sua prima 'macchina differenziale' (1822).

```
10 REM  ************************
    ********
20 REM  **
30 REM  **
40 REM  **   ALBERTO  &  ANDREA
50 REM  **
60 REM  **          BORIANI
70 REM  **
75 REM  **   ASTA CHE CADE  PER
    C64   **
80 REM  **
85 REM  ************************
    ********
90 REM  **  QUESTO PROGRAMMA UTI
    LIZZA **
92 REM  ** LE NUOVE ROUTINES GRA
    FICHE **
93 REM  ** DI DANILO TOMA (C.C.C
     N14) **
95 REM  ************************
    ********
100 G=9.8/6:N=20:PX=-140
104 PRINT"[CLEAR]PRIMA ESPERIENZA
    :":GOSUB  500
105 REM  ************************
    *******
106 REM  **       ASTA ORIZZONTALE
         **
107 REM  ************************
    *******
110 ←CLEAR:←GRAF 6,3:←COL OR  1
115 GOSUB  283
120 FOR  T=0  TO  11  STEP  .5
130 XG=18*T-119:YG=-1.1*T*T+50
140 X1=XG-30:X2=XG+30
150 ←COL OR  0:←DRAW $,$,$,$,$,$
155 ←COL OR  1:←PLOT PX,PY,0
160 ←DRAW X1,YG,0,X2,YG,0
165 PX=X1:PY=YG
170 FOR  R=1  TO  50:NEXT R,T
180 GET  A$:IF  A$=""  THEN  180
190 PRINT"[CLEAR]"
195 ←TEXT 6,14:←CLEAR:PRINT"[CLEA
    R]SECONDA ESPERIENZA:":GOSUB
     560
196 REM  ************************
    *******
197 REM  **       ASTA VERTICALE
         **
198 REM  ************************
200 ←GRAF 6,3:GOSUB  283
210 FOR  T=0  TO  11  STEP  .5
220 X1=18*T-119:YG=-1.1*T*T+67
230 Y1=YG-30:Y2=YG+30
235 ←COL OR  0:←DRAW $,$,$,$,$,$
237 ←COL OR  1:←PLOT PX,PY,0
240 ←DRAW X1,Y1,0,X1,Y2,0
245 PX=X1:PY=Y1
250 FOR  F=1  TO  50:NEXT F
260 NEXT T
270 GET  A$:IF  A$=""  THEN  270
275 PRINT"[CLEAR]"
277 REM  ************************
278 REM  **        MURO & TERRA
         **
279 REM  ************************
280 ←TEXT 6,14:←CLEAR:GOSUB  610:
    ←GRAF 6,3
281 GOTO 310
283 FOR  A=-150  TO  -108:←COL OR
      1:←DRAW A,33,0,A,-99,0:NEXT
285 FOR  A=-99  TO  -83  STEP 2:←
    DRAW -107,A,0,150,A,0:NEXT
287 FOR  A=-99  TO  33  STEP  6:←
    COL OR 0
288 ←DRAW -150,A,0,-108,A,0: NEXT
290 FOR  A=-99  TO  33  STEP 12:←
    COL OR 0
292 ←DRAW -140,A-12,0,-140,A-6,0
294 ←DRAW -120,A-12,0,-120,A-6,0
300 ←DRAW -130,A-6,0,-130,A,0: NE
    XT
305 RETURN
307 REM  ************************
    *******
308 REM  **        ASTA ROTANTE
         **
309 REM  ************************
    *******
310 GOSUB  283:PX=-120:PY=0:T=0
315 T=T+.19
320 XG=N*T-119:YG=-G*T*T+64
330 X=30*SIN (T):Y=30*COS(T)
340 ←COL OR  0:←DRAW $,$,$,$,$,$
350 ←COL OR  1:←PLOT PX,PY,0
360 ←DRAW XG+X,YG+Y,0,XG-X,YG-Y,0
370 PX=XG+X:PY=YG+Y
380 IF  PY>-83  AND  PX<159  AND
     (XG-X)<159  AND  (YG-Y)>-83
     THEN 315
```

```
390 GET  A$:IF  A$=""  THEN  390
410 T=0
415 T=T+.1
420 XG=N*T-119:YG=-G*T*T+64
455 ←COL OR  1:←PLOT XG,YG,0
480 IF  YG>-83  AND  XG<159  THEN
     415
490 GET  A$:IF  A$=""  THEN  490
495 REM
497 GOTO 1000
500 PRINT"SIMULAZIONE DI UN'ASTA
    ORIZZONTALE CHE CADE"
505 PRINT"DX=V*T   (MOTO RETT. UN
    IF.)"
510 PRINT"DY=G*T↑2 (CORPO IN CADU
    TA LIBERA)"
515 PRINT"(PREMI UN TASTO PER CON
    TINUARE)"
516 PRINT"(PREMI UN TASTO ALLA FI
    NE DEL DISEGNO)"
520 GET  A$:IF  A$=""  THEN  520
530 RETURN
560 PRINT" SIMULAZIONE DI UN'ASTA
     VERTICALE CHE   CADE"
565 PRINT"DX=V*T   (MOTO RETT. UN
    IF.)"
570 PRINT"DY=G*T↑2 (CORPO IN CADU
    TA LIBERA)"
575 PRINT"(PREMI UN TASTO)"
580 GET  A$:IF  A$=""  THEN  580
590 RETURN
610 PRINT"SIMULAZIONE DI UN'ASTA
    CHE CADE RUOTANDO INTORNO AL
    SUO";
612 PRINT" CENTRO"
615 PRINT"DX=V*T   (MOTO RETT. UN
    IF.)"
620 PRINT"DY=G*T↑2 (CORPO IN CADU
    TA LIBERA)"
625 PRINT" PREMENDO UN TASTO VIEN
    E TRACCIATA LA    TRAIETTORIA
     DEL SUO CENTRO"
630 GET  A$:IF  A$=""  THEN  630
640 RETURN
1000 PRINT"[CLEAR]TERZA ESPERIENZZ
     A":←TEXT 6,14
1010 PRINT" VELOCITA' (V) ";
1020 INPUT  N
1030 INPUT " GRAVITA' (G) ";G
1040 ←CLEAR:←GRAF 6,3:GOTO 307
```

ISOMERI

Visualizzare sullo schermo la struttura molecolare di una formula qualsiasi, che essa esista in natura oppure no.

Permette anche di vedere in quanti modi possibili gli atomi di una data sostanza organica possono legarsi tra loro, di veder cioè rappresentati i cosiddetti isomeri di un dato composto chimico.

Chimica, chimica!

Chimica organica, di che si tratta? Il nome porta a pensare che si tratti di una parte della chimica che tratta in particolare le sostanze organiche, cioè quelle sostanze aventi una stretta relazione con gli organismi animali e vegetali. Anche l'esperienza sembrerebbe confermare questa idea. Molte sostanze, dette organiche, derivano da organismi animali e vegetali e sono di importanza fondamentale per l'attività vitale degli organismi stessi.

Ma sarebbe del tutto inesatto ammettere che le sostanze organiche si trovino esclusivamente negli organismi, o che tutte le sostanze in cui si riscontrano siano di natura organica. In realtà si intendono oggi, come sostanze organiche, tutte quelle sostanze nella composizione delle quali sia presente il carbonio, indipendentemente dal modo di produrre o di estrarre quelle sostanze.
Così, i termini "chimica organica" e "sostanze organiche" appaiono fondamentalmente inesatti. E' più esatto definire la chimica organica come la chimica dei composti a base di carbonio.

Il carbonio, come qualsiasi altro elemento chimico, può essere considerato come il singolo pezzetto di un puzzle. Per l'esattezza noi considereremo tre diversi elementi chimici, ossia tre diversi tipi di pezzetti di un puzzle: il carbonio, l'idrogeno e l'ossigeno.

Questi tre "pezzetti" hanno caratteristiche diverse e naturalmente la caratteristica fondamentale del pezzetto di un puzzle è di potersi unire ad altri pezzetti. L'atomo di idrogeno ha la caratteristica di potersi unire ad un solo altro atomo, quindi nel nostro puzzle andrà a costituire una zona periferica: potrà cioè essere l'ultimo anello di una catena di pezzetti. L'atomo di ossigeno, invece, può legarsi ad altri due pezzetti oppure se preferite può legarsi "doppiamente" ad un altro. L'ossigeno quindi può fare da "ponte" fra due pezzetti. Il carbonio invece (nei composti che trattiamo noi) può legarsi con ben quattro altri atomi e possiede anche la proprietà eccezionale di poter unire in catena continua o ramificata molti dei suoi atomi, dando origine ad un numero illimitato di prodotti.

Per meglio capire quanto detto sopra, osservate la figura 1 dove potete vedere un atomo di carbonio al centro e quattro atomi di idrogeno legati ad esso: il composto così ottenuto si chiama Metano ed è il gas con il quale cuocete la pasta asciutta. Nella figura 2 invece potete vedere due atomi di carbonio legati in catena ed attorniati da sei atomi di idrogeno. Il tutto si chiama Etano, mentre se avessimo collegato in catena tre atomi di carbonio avremo ottenuto il Propano, con quattro atomi di carbonio il Butano e così via.

Spesso succede però che due atomi non si leghino tra loro con un solo legame, ma con due o tre legami: come nel caso dell'Acetilene, che vedete rappresentato in figura 3. Fino ad ora abbiamo analizzato casi semplici, ma provate a rappresentare un composto costituito da tre atomi di

carbonio, otto atomi di idrogeno ed un atomo di ossigeno. Scoprirete coaì che esiste più di un modo per farlo.

Con lo stesso numero e tipo di atomi potete infatti costruire ad esempio tre "puzzles" diversi. Ad esempio l'Alcool Propilico di figura 4 che ha la stessa composizione dell'Alcool Isopropilico di figura 5 e dell'Etere Metil-etilico di figura 6. Questi particolari composti con medesima composizione chimica si chiamano Isomeri.

La parola "isomeria" introdotta nel 1830 dal chimico Berzelius, indicava composti aventi uguale composizione, ma proprietà diverse. La chimica moderna invece preferisce utilizzare la parola "isomeri" per i composti che hanno composizione chimica e proprietà uguali, mentre utilizza la parola "metameri" per i composti che hanno composizione chimica uguale ma proprietà differenti. Per la chimica moderna, quindi, i composti di figura 4 e di figura 5 sono tra loro isomeri, mentre i composti di figura quattro e di figura 6 sono tra loro metameri. Noi, affezionati a Berzelius, continueremo ad usare il termine isomeri per entrambi i casi commettendo così un'imprecisione.

Potete quindi ben immaginare ora quanti isomeri può avere un composto costituito da sei atomi di carbonio, dodici atomi di idrogeno e sei atomi di ossigeno come il Glucosio, zucchero che tutti noi abbiamo nel sangue.

Potete altresì capire quanto difficile sia rappresentare in forma grafica tutte queste molecole, ma non preoccupatevi: questo programma potrà aiutarvi a farlo.

Il programma

Il programma in questione disegna in pagina grafica la cosiddetta "formula di struttura" di un qualsiasi composto costituito da un massimo di dodici atomi di carbonio, ventisei atomi di idrogeno e sei atomi di ossigeno. Naturalmente, per disegnare in alta risoluzione è necessario far uso delle ottime nuove routines grafi-

che di Danilo Toma pubblicate sul numero quattordici di Commodore Computer Club. Caricate quindi le nuove routines di Toma, lanciatele, e poi caricate in memoria il programma Isomeri.

Se avete eseguito tutto questo correttamente, il programma vi chiederà quanti atomi di carbonio sono presenti nel composto che volete vedere, seguirà poi la stessa domanda per quanto riguarda l'idrogeno e l'ossigeno. Se i dati da voi digitati escono dai limiti prefissati il computer segnerà l'errore e vi ripeterà la domanda, altrimenti proseguirà chiedendovi la dimensione della molecola (perchè alcune molecole più complesse potrebbero uscire dallo schermo, mentre altre potrebbero risultare troppo piccole e poco leggibili).

A questo punto seguirà una pausa più o meno lunga a seconda della complessità della molecola che avete richiesto: la pausa sarà brevissima per un composto come il Metano o l'Acetilene, mentre sarà un po' più consistente nel caso ad esempio, del Glucosio. Una volta terminata l'elaborazione che consiste principalmente in un riempimento di varie matrici in modo pseudocausale, il computer passerà in alta risoluzione e disegnerà la molecola richiesta.

Al disegno della molecola seguirà la stampa in alta risoluzione di una breve dicitura che vi indica la possibilità di richiedere un altro isomero della stessa molecola, di vedere la stessa molecola con un'altra angolazione oppure di tornare all'inizio del programma. Provate ora ad inserire la richiesta della molecola del Metano rispondendo "1" alla richiesta del numero di atomi di carbonio, "4" per il numero di atomi di idrogeno e "0" per l'ossigeno.

Pigiate "4" per la dimensione della molecola e quasi immediatamente comparirà il disegno della formula di struttura del Metano dove il quadratino pieno rappresenta l'atomo di carbonio ed i quadratini vuoti gli atomi di idrogeno, mentre se fossero presenti atomi di ossigeno verrebbero rappresentati come cerchietti.

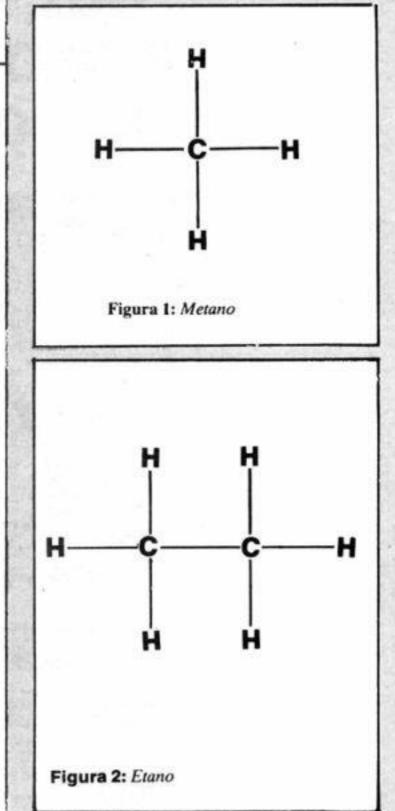
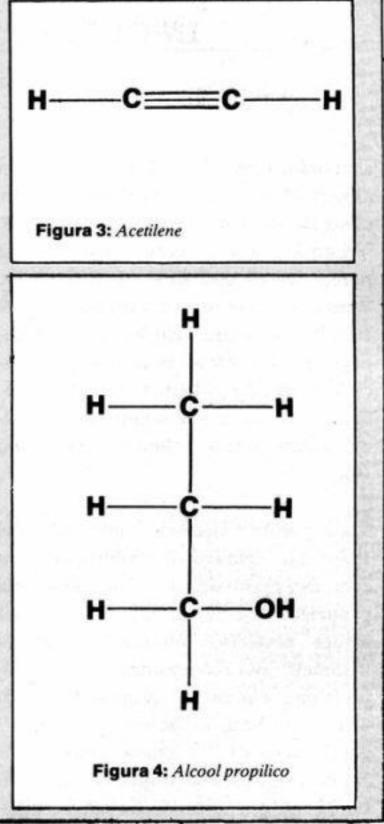



Figura 1: *Metano*

Figura 2: *Etano*

Figura 3: *Acetilene*

Figura 4: *Alcool propilico*

Naturalmente, se a questo punto richiedete un altro isomero il risultato sarà sempre lo stesso poichè non possono esistere isomeri della molecola di metano. Lo stesso effetto nullo avrà la richiesta di un'altra angolazione e questo per il motivo evidente che la molecola di Metano è simmetrica. Inserite ora la richiesta per la molecola di Acetiline con due atomi di carbonio, due di idrogeno e zero di ossigeno ed otterrete il corrispondente di figura 3.

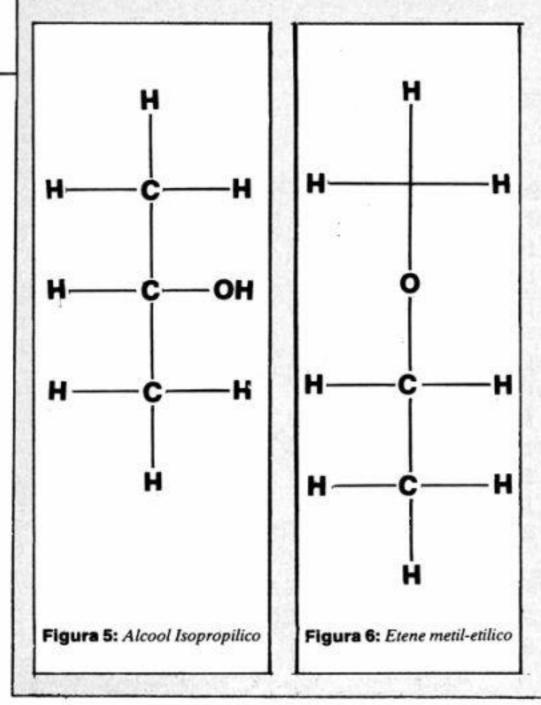
Non spaventatevi se talvolta gli atomi di idrogeno appariranno come aventi due legami, questa è solo un'illusione data dalla bi-dimensionalità della figura. Infatti l'idrogeno si trova solo sovrapposto ad un altro legame. Talvolta potrebbero apparire righe senza senso apparente: questo fenomeno è dovuto alla eccessiva complessità della molecola richiesta. Potete ora provare a richiedere un isomero (o metamero se preferite) della molecola del Glucosio rispondendo "6" per il numero di atomi di carbonio, "12" per gli atomi di idrogeno e "6" per quelli di ossigeno. Otterrete cosi' una molecola che non ha niente a che fare con la molecola di Glucosio che trovate sui libri di chimica, e questo perchè esiste un numero grandissimo di isomeri di questo composto.

Infatti, se provate a premere "I" per richiedere un altro isomero potete ottenere una molecola completamente diversa dalla prima che però mantiene invariato il numero e il tipo di atomi. Non sperate troppo di ottenere la molecola reale del Glucosio continuando a richiedere isomeri o altre angolazioni perchè dovreste essere molto fortunati per poterlo fare!

Se però avete qualche anno libero che non sapete come utilizzare, potete sempre provare a contare il numero di isomeri possibili di una molecola con dodici atomi di carbonio, ventisei di idrogeno e sei di ossigeno. Scrivetemi per qualsiasi problema di malfunzionamento del programma, che per altro ha già funzionato egregiamente in versioni diverse anche su personal computer differenti dal buon vecchio 64.

Luca Galuzzi

Figura 5: *Alcool Isopropilico*

Figura 6: *Etene metil-etilico*

```
100 REM ************************
    ******
120 REM *         I S O M E R I
    *
130 REM *
    *
140 REM *         COMMODORE  64
    *
160 REM *         LUCA  GALUZZI
    *
230 REM *
    *
240 REM ************************
    ******
250 REM * IL PROGRAMMA RICHIEDE
    L'USO*
260 REM * DELLE NUOVE ROUTINES G
    RAFI-*
270 REM * CHE DI D.TOMA (#14 C.C
    .C.) *
280 REM ************************
    ******
290 DIM  CP(15,2),OP(6,2),HP(34,2
    ),C(15,4),O(6,2)
300 ←TEXT 0,11:POKE  53280,0:PRIN
    T "[CLEAR]"
310 INPUT  "NUMERO ATOMI DI CARBO
    NIO ";NC%
320 INPUT  "NUMERO ATOMI DI IDROG
    ENO ";NH%
330 INPUT  "NUMERO ATOMI DI OSSIG
    ENO ";NO%
340 INPUT  "DIMENSIONI DISEGNO (2
    ,3,4)";V%
350 IF  NC%<1  OR  NH%<2  OR  NO%
    <0  OR  NC%>15  OR  NO%>6  TH
    EN  380
360 IF  INT(NH%/2)<>NH%/2  OR  NH
    %>2*NC%+2  THEN  380
370 GOTO  400
380 PRINT "  CONTROLLARE I DATI I
    N INGRESSO        ":REM  7 SP
390 PRINT " 0=< C <16 , 2=< H <=2
    *C+2 , 0=< O <7 ":GOTO  310
400 IF  V%<2  OR  V%>4  THEN  340
410 V%=V%*10
420 EU=((2*NC%+2)-NH%)/2
430 FOR  I=1  TO  NC%
440 FOR  J=1  TO  4:C(I,J)=0:NEXT
     J,I
450 FOR  I=1  TO  NO%
460 FOR  J=1  TO  2:O(I,J)=0:NEXT
     J,I
470 HU=0:OU=0:UN=0:S1=130:S2=70
480 IF  NC%=1  THEN  570
490 C(1,1)=2:C(2,1)=1:IF  NC%=2
    THEN  570
500 FOR  I=3  TO  NC%
510 X=INT(RND(1)*(I-1))+1
520 IF  C(X,2)=0  THEN  C(I,1)=X:
    C(X,2)=I:GOTO  560
530 IF  C(X,3)=0  THEN  C(I,1)=X:
    C(X,3)=I:GOTO  560
540 IF  C(X,4)=0  THEN  C(I,1)=X:
    C(X,4)=I:GOTO  560
550 GOTO  510
560 NEXT I
570 IF  NO%=0  THEN  690
580 FOR  I=1  TO  NO%
590 X=INT(RND(1)*(NC%+I-1))+1
600 IF  X<=NC%  THEN  630
610 GOTO  630
620 O(X-NC%,2)=-I:O(I,1)=-(X-NC%)
    :GOTO  680
630 IF  X<=NC%  THEN  640
640 IF  C(X,2)=0  THEN  C(X,2)=-I
    :O(I,1)=X:GOTO  680
650 IF  C(X,3)=0  THEN  C(X,3)=-I
    :O(I,1)=X:GOTO  680
660 IF  C(X,4)=0  THEN  C(X,4)=-I
    :O(I,1)=X:GOTO  680
670 GOTO  590
680 NEXT I
```

```
690 IF  UN=EU  THEN  850
700 IF  NC%=1  THEN  750
710 GOSUB  1060
720 IF C(X1,X2)=0 AND C(Y1,Y2)=0
    THEN 740
730 GOTO  750
740 C(X1,X2)=Y1:C(Y1,Y2)=X1:UN=UN
    +1:IF  UN=EU  THEN  850
750 IF  NO%=0  THEN  710
760 GOSUB  1120
770 IF  C(X1,X2)=0  AND  O(Y1,2)=
    0  THEN  790
780 GOTO  800
790 C(X1,X2)=-Y1:O(Y1,2)=X1:UN=UN
    +1:GOTO  690
800 IF  NO%=1  THEN  690
810 GOSUB  1160
820 IF  O(X1,2)=0  AND  O(Y1,2)=0
     THEN  840
830 GOTO  850
840 O(X1,2)=-Y1:O(Y1,2)=-X1:UN=UN
    +1:GOTO  690
850 IF  NC%=1  THEN  C(1,1)=-101:
    HU=HU+1
860 FOR  I=1  TO  NC%
870 FOR  J=2  TO  4
880 IF  C(I,J)=0  THEN  C(I,J)=-(
    101+HU):HU=HU+1
890 NEXT J,I
900 IF  NO%=0  THEN  950
910 FOR  I=1  TO  NO%
920 IF  O(I,2)=0  THEN  O(I,2)=-(
    101+HU):HU=HU+1
930 NEXT I
940 IF  NH%<>HU  THEN  430
950 GOSUB  1200
960 GOSUB  1950
970 GOSUB  2900
980 GET  A$:IF  A$=""  THEN  980
990 IF  A$="A"  THEN  950
1000 IF  A$="I"  THEN  430
1010 IF  A$="R"  THEN  300
1020 GOTO  980
1030 REM  ************************
     *******
1040 REM  * CALCOLA I PARAMETRI CA
     SUALI *
1050 REM  ************************
     *******
1060 X1=INT(RND(1)*NC%)+1
1070 Y1=INT(RND(1)*NC%)+1
1080 IF  X1=Y1  THEN  1060
1090 X2=INT(RND(1)*3)+2
1100 Y2=INT(RND(1)*3)+2
1110 RETURN
1120 X1=INT(RND(1)*NC%)+1
1130 Y1=INT(RND(1)*NO%)+1
1140 X2=INT(RND(1)*3)+2
1150 RETURN
1160 X1=INT(RND(1)*NO%)+1
1170 Y1=INT(RND(1)*NO%)+1
1180 IF  X1 = Y1  THEN  1160
1190 RETURN
1200 FOR  I=1  TO  NC%:FOR  J=0  T
     O  2
1210 CP(I,J)=0:NEXT J,I
1220 IF NO%=0 THEN 1260
1230 FOR  I=1  TO  NO%:FOR  J=0  T
     O  2
1240 OP(I,J)=0
1250 NEXTJ,I
1260 FF=0:CP(1,1)=120:CP(1,2)=75
1270 CP(1,1)=120
1280 FOR  II=1  TO  NC%
1290 IF  CP(II,0)=0  THEN  1320
1300 NEXT II
1310 GOTO  1610
1320 FOR  I=II  TO  NC%
1330 IF  CP(I,0)=1  OR  CP(I,1)=0
      THEN  1590
1340 XX=INT(RND(1)*4)
1350 FOR  J=1  TO  4
1360 KK=0
1370 IF  XX=0  THEN  X=CP(I,1)+V%:
     Y=CP(I,2)
1380 IF  XX=1  THEN  X=CP(I,1):Y=C
     P(I,2)-V%
1390 IF  XX=2  THEN  X=CP(I,1)-V%:
     Y=CP(I,2)
1400 IF  XX=3  THEN  X=CP(I,1):Y=C
     P(I,2)+V%
1410 XX=XX+1
1420 IF  XX=4  THEN  XX=0
1430 IF  C(I,J) >= -100  THEN  146
     0
1440 HP(ABS(C(I,J)+100),1)=((X-CP(
     I,1))*7/20)+CP(I,1)
1450 HP(ABS(C(I,J)+100),2)=((Y-CP(
     I,2))*7/20)+CP(I,2):GOTO  157
     0
1460 TX=X::TY=Y:FF=0:GOSUB 1810:IF
      FF=-1  AND  KK<3  THEN  KK=
```

```
     KK+1:GOTO  1370
1470 IF  FF=-1  AND  KK>2  THEN  1
     200
1480 IF  NO%=0  THEN  1510
1490 IF  C(I,J)>0  THEN  1540
1500 IF  C(I,J)<0  AND  C(I,J)>-10
     0  AND  OP(ABS(C(I,J)),1)>0
     THEN  1570
1510 IF  C(I,J)<0  AND  C(I,J)>-10
     0  THEN  1530
1520 GOTO  1540
1530 OP(ABS(C(I,J)),1)=X:OP(ABS(C(
     I,J)),2)=Y:GOTO  1570
1540 IF  CP(C(I,J),1)>0  THEN  157
     0
1550 CP(C(I,J),1)=X
1560 CP(C(I,J),2)=Y
1570 NEXT J
1580 CP(I,0)=1
1590 NEXT I
1600 GOTO  1280
1610 IF  NO%=0  THEN  1800
1620 FOR  I=1  TO  NO%
1630 IF  OP(I,1)>0  THEN  1730
1640 FOR  J=1  TO  NO%
1650 IF  J=I  THEN  1720
1660 IF  O(J,1)<>-I  THEN  1720
1670 OP(I,1)=OP(J,1)
1680 OP(I,2)=OP(J,2)+20
1690 TX=OP(I,1):TY=OP(I,2):FF=1
1700 GOSUB  1810
1710 IF  FF=-1  THEN  1200
1720 NEXT J
1730 IF  O(I,1)>=-100  THEN  1760
1740 HP(ABS(O(I,1)+100),1)=OP(I,1)
     +7
1750 HP(ABS(O(I,1)+100),2)=OP(I,2)
1760 IF  O(I,2)>=-100  THEN  1790
1770 HP(ABS(O(I,2)+100),1)=OP(I,1)
     -7
1780 HP(ABS(O(I,2)+100),2)=OP(I,2)
1790 NEXT I
1800 RETURN
1810 FOR  K=1  TO  NC%
1820 IF  K=C(I,J)  AND  FF=0  THEN
      1840
1830 IF  TX=CP(K,1)  AND  TY=CP(K,
     2)  THEN  FF=-1:GOTO  1910
1840 NEXT K
1850 IF  NO%=0  THEN  1910
1860 FOR  K=1  TO  NO%
1870 IF  K=ABS(C(I,J))  AND  FF=0
      THEN  1900
1880 IF  K=I  AND  FF=1  THEN  190
     0
1890 IF  TX=OP(K,1)  AND  TY=OP(K,
     2)  THEN  FF=-1:GOTO  1910
1900 NEXT K
1910 RETURN
1920 REM  ************************
     **
1930 REM  *  DISEGNA LA MOLECOLA
      *
1940 REM  ************************
     **
1950 ←CLEAR
1960 ←GRAF 0,1
1970 ←COLOR  1
1980 FOR  I=1  TO  NC%
1990 FOR  J=1  TO  4
2000 IF  C(I,J)>=-100  THEN  2050
2010 G1=HP(ABS(C(I,J)+100),1)-S1
2020 G2=HP(ABS(C(I,J)+100),2)-S2
2030 ←DRAW CP(I,1)-S1,CP(I,2)-S2,0
     ,G1,G2,0
2040 GOTO 2350
2050 IF  C(I,J)>0  THEN  2180
2060 FF=0:FOR  K=1  TO  4
2070 IF  K=J  THEN  2090
2080 IF  C(I,K)=C(I,J)  THEN  FF=-
     1
2090 NEXT K
2100 G1=OP(ABS(C(I,J)),1)-S1
2110 G2=OP(ABS(C(I,J)),2)-S2
2120 ←DRAW CP(I,1)-S1,CP(I,2)-S2,0
     ,G1,G2,0
2130 IF  FF<>-1  THEN  2170
2140 G1=OP(ABS(C(I,J)),1)+3-S1
2150 G2=OP(ABS(C(I,J)),2)+3-S2
2160 ←DRAW CP(I,1)+3-S1,CP(I,2)+3-
     S2,0,G1,G2,0
2170 GOTO  2350
2180 FF=0:IF  C(I,J)<I  THEN  2350
2190 IF C(I,J)<I THEN 2350
2200 FOR  K=1  TO  4
2210 IF  K=J  THEN  2230
2220 IF  C(I,J)=C(I,K)  THEN  FF=F
     F+1
2230 NEXT K
2240 G1=CP(C(I,J),1)-S1
2250 G2=CP(C(I,J),2)-S2
2260 ←DRAW CP(I,1)-S1,CP(I,2)-S2,0
```

```
      ,G1,G2,0
2270 IF  FF<=0  THEN  2310
2280 G1=CP(C(I,J),1)+3-S1
2290 G2=CP(C(I,J),2)+3-S2
2300 ←DRAW CP(I,1)+3-S1,CP(I,2)+3-
      S2,0,G1,G2,0
2310 IF  FF<>2  THEN  2350
2320 G1=CP(C(I,J),1)-3-S1
2330 G2=CP(C(I,J),2)-3-S2
2340 ←DRAW CP(I,1)-3-S1,CP(I,2)-3-
      S2,0,G1,G2,0
2350 NEXT J
2360 NEXT I
2370 IF  NO%=0  THEN  2520
2380 FOR  I=1  TO  NO%
2390 FOR  J=1  TO  2
2400 IF  O(I,J)>0  THEN  2500
2410 IF  O(I,J)>=-100  THEN  2460
2420 G1=HP(ABS(O(I,J)+100),1)-S1
2430 G2=HP(ABS(O(I,J)+100),2)-S2
2440 ←DRAW OP(I,1)-S1,OP(I,2)-S2,0
      ,G1,G2,0
2450 GOTO  2500
2460 IF  O(I,J)>=0  THEN  2500
2470 G1=OP(ABS(O(I,J)),1)-S1
2480 G2=OP(ABS(O(I,J)),2)-S2
2490 ←DRAW OP(I,1)-S1,OP(I,2)-S2,0
      ,OP(ABS(O(I,J)),1),OP(ABS(O(I
      ,J)),2),0
2500 NEXT J
2510 NEXT I
2520 QQ=1:FOR  I=1  TO  NC%
2530 FOR I=1 TO NC%
2540 X=CP(I,1)
2550 Y=CP(I,2)
2560 GOSUB  2750
2570 NEXT I
2580 IF  NO%=0  THEN  2650
2590 QQ=2:FOR  I=1  TO  NO%
2600 FOR I=1 TO NO%
2610 X=OP(I,1)
2620 Y=OP(I,2)
2630 GOSUB  2750
2640 NEXT I
2650 QQ=3:FOR  I=1  TO  NH%
2660 FOR I=1 TO NH%
2670 X=HP(I,1)
2680 Y=HP(I,2)
2690 GOSUB  2750
2700 NEXT I
2710 RETURN
2720 REM  ************************
      ***
2730 REM  * DISEGNA I SINGOLI ATOM
      I *
2740 REM  ************************
      ***
2750 IF  QQ<>1  THEN  2790
2760 FOR  L=Y-3  TO  Y+3
2770 ←DRAW X-3-S1,L-S2,0,X+3-S1,L-
      S2,0
2780 NEXT L
2790 IF  QQ<>2  THEN  2810
2800 ←CIRCLE X-S1,Y-S2,0,4,4
2810 IF  QQ<>3  THEN  RETURN
2820 ←DRAW X-2-S1,Y+2-S2,0,X+2-S1,
      Y+2-S2,0
2830 ←DRAW X+2-S1,Y+2-S2,0,X+2-S1,
      Y-2-S2,0
2840 ←DRAW X+2-S1,Y-2-S2,0,X-2-S1,
      Y-2-S2,0
2850 ←DRAW X-2-S1,Y-2-S2,0,X-2-S1,
      Y+2-S2,0
2860 RETURN
2870 REM  ************************
      *****
2880 REM  *SCRIVE IN ALTA RISOLUZI
      ONE *
2890 REM  ************************
      *****
2900 PRINT "[CLEAR]I - ISOMERO"
2910 PRINT "A - ALTRA ANGOLAZIONE"
2920 PRINT "R - RITORNA"
2930 J=11:K=0:GOSUB  2970
2940 J=21:K=40:GOSUB  2970
2950 J=11:K=80:GOSUB  2970
2960 RETURN
2970 POKE  56334,PEEK(56334)  AND
      254
2980 POKE  1,PEEK(1)  AND  251
2990 P=1024:PP=8
3000 KK=53248:JJ=57344
3010 FOR  R=K  TO  J+K
3020 CH=PEEK(P+R)
3030 FOR  T=1  TO  PP
3040 CG=PEEK(KK+CH*PP+(T-1))
3050 POKE  JJ+R*PP+(T-1),CG
3060 NEXT T,R
3070 POKE  1,PEEK(1)  OR  4
3080 POKE  56334,PEEK(56334)  OR
      1
3090 RETURN
```

eseguono una somma per ciascuna lettura dei DATA. Supponiamo che il programma (solo dimostrativo: non serve a nulla) sia il seguente:

```
100 FOR I = 1 TO 10
110 READ X: POKE 10000=I,X
120 NEXT
130 DATA 1,2,3,4,5,6,7,8,9,10
```

Se nella riga 130 invece di scrivere il numero 8 scrivi, ad esempio, il numero 9, potresti provocare gravi malfunzionamenti nell'esecuzione del programma. Per evitare ciò si introducono un paio di righe di controllo simili alle seguenti:

```
99 A=0
111 A=A+X
125 IF A< >55 THEN
PRINT"ERRORE":END
```

La somma dei dieci valori è, infatti, pari a 55 ed un eventuale errore di digitazione non "passa" attraverso la riga 125.

E' ovvio che questo sistema non fornisce una garanzia al 100% nei casi in cui, ad esempio, si commettono due errori di trascrizione l'uno in eccesso e l'altro in difetto. Il sistema descritto serve, comunque, solo per consentire ai lettori di riviste di digitare correttamente i listati pubblicati. Una volta corretti eventuali errori, le linee di controllo possono benissimo essere cancellate.

C-64 Parlante

☐ **Sono in possesso del programma Sam in grado di far parlare il C-64 ma non sono in grado di farlo funzionare. Potreste aiutarmi?** *(Massimo Gallinaro - Padova)*

• Non possediamo il libretto di istruzioni del programma che citi. Hai provato, comunque, a "sentire" la cassetta che la rivista COMMODORE CLUB (speciale L. 10000) ha pubblicato col titolo "LA VOCE"? Questa fa parlare il tuo C-64 in italiano!

Monitor b/n Commodore

☐ **Perchè la Commodore non commercializza monitor in bianco-nero?** *(Massimo Nitri - Cinisello Balsamo)*

• E' probabile che i motivi siano solo commerciali (= guadagnerebbe di meno...) oppure perchè sarebbe un controsenso vendere monitor b/n quando la produzione è relativa a computer con uscita a colori...

Da nastro a disco

☐ **Come fare per trasferire su disco i file di un programma che utilizzava il registratore? Nonostante i molti tentativi il drive lampeggia continuamente!** *(Maurizio Gregori - Pesaro)*

• Supponiamo che i file siano correttamente aperti secondo le direttive indicate nel libretto d'istruzioni (cioè [S,W] per scrivere e [S,R] per leggere) e che non vi siano confusioni tra file aperti per leggere usati per scrivere e viceversa. L'errore che più comunemente si commette consiste nell'aggiornamento del file. Se, in altre parole, carichiamo da disco il file denominato ARCHIVIO (registrato in precedenza), ne elaboriamo alcuni dati e tentiamo di riscrivere il file così modificato con lo stesso nome "ARCHIVIO", il drive si rifiuterà di eseguire l'ordine perchè sullo stesso disco esiste già un file (quello "vecchio") con lo stesso nome.

Le operazioni da effettuare per aggiornare un file su disco sono dunque le seguenti:

• Caricare in memoria l'intero file.

• Effettuare le modifiche che si desiderano.

• Cancellare il file "vecchio" su disco (mediante la sintassi per effettuare lo "scratch").

• Scrivere il file elaborato dal computer con lo stesso nome di prima.

In effetti si può ricorrere alla particolare sintassi che, utilizzando il carattere chiocciolina ( ), permette di scrivere il file nuovo su quello vecchio dotato dello stesso nome. Il ricorso a tale sistema, però, provoca a lungo andare vari inconvenienti nella Directory del disco.

Se, comunque, il drive lampeggia e non sai il perchè, ciò è dovuto alla superficie con cui hai scritto il programma. Ricorrendo, infatti, ai consigli riportati sul manuale, avresti dovuto inserire la subroutine che consente di effettuare la verifica dopo ciascuna operazione di lettura o scrittura. Per esser più chiari, dopo ogni istruzione di tipo GET , INPUT , PRINT è indispensabile eseguire una subroutine del tipo che segue:

```
60000 INPUT  15, A,B$,C,D
60010 IF A< > 0 THEN PRINT
A,B$,C,D: END
60020 RETURN
```

Un valore di "A" diverso da zero, infatti, indica malfunzionamenti nel "colloquio" col drive. E' ovvio che prima di accedere alla subroutine è indispensabile aprire il file di controllo N.15. Nel caso esaminato il computer avrebbe risposto FILE EXIST e ti sarebbe stato più semplice individuare l'errore.

Linguaggio macchina

☐ **C'è qualche testo che permetta di avvicinarsi al mondo del L.M.? Possiedo alcuni monitor L.M., ma non so come usarli.** *(Vincenzo Arcidiacono - Milano)*

• Affrontiamo sistematicamente, da un po' di tempo, argomenti concernenti il L.M., il Kernal, ed altre cose del genere. Tra un po' presenteremo un'iniziativa di cui sarai certamente soddisfatto. Per il

EASY COMPUTING
VIA A.BERTANI 24 FIRENZE

momento, purtroppo, accontentati di ciò che vedi sulle nostre pagine.

Fotografie col C-64

☐ **Sul N. 15 di C.C.C. affermate che è possibile, ricorrendo ad una telecamera ed alla pagina grafica in alta risoluzione, digitalizzare una fotografia col C-64 in modo da poterla "richiamare" a volontà, per esempio per fondali di videogiochi. E' possibile questo collegamento con la telecamera di un videoregistratore?** *(Massimiliano Biancofiore - Bari)*

• Purtroppo la sola telecamera non è sufficiente. E' necessario un apparecchio in grado di effettuare lo scanning (esame riga per riga) dell'immagine ripresa ed il conseguente trasferimento nella memoria del computer. Se qualche lettore conosce il nome di una Ditta che commercializza un prodotto del genere saremo ben lieti di pubblicarlo. Dubitiamo, però, che l'operazione risulti economicamente affrontabile.

Pezzi di ricambio

☐ **Dopo 20 giorni dall'acquisto (gennaio 84 non 85) la testina di scrittura della mia sstampante MPS801 si è rotta e finora non è stata riparata a causa della mancanza del pezzo di ricambio. Non pensate che sia una vergogna?** *(Vincenzo Serotti - Firenze)*

• Sì. Non nascondo che, in casi come il tuo, mi piacerebbe vedere del rossore (anche su una sola guancia). Solo per verificare la presunta appartenenza alla specie umana dei responsabili di taluni centri di assistenza.

Ancora libri

☐ **Potreste consigliarmi alcuni libri per sopperire alla carenze del manuale di cui è dotato il mio Commodore 64?** *(Antonio Leonetti - Matera)*

• Tra breve diffonderemo, come numero speciale della rivista su cassetta Commodore Club, un corso di Basic proprio per venire incontro ai principianti. Ti consiglio, comunque, di seguire con attenzione tutti gli articoli che pubblichiamo e che, come puoi notare, sono in grado di soddisfare sia gli esperti sia coloro che, come te, possiedono da poco un calcolatore Commodore.

Aprire un club

☐ **Siamo un gruppo di amici e avremmo intenzione di aprire un club di utenti del Commodore 64. Potreste aiutarci?** *(Lettera firmata)*

• Tempo fa affermammo che abbiamo intenzione di aiutare, con consigli ed altre iniziative, i giovani intenzionati ad aprire seriamente un club. Purtroppo ci siamo accorti che dietro la richiesta di pubblicare nominativi (ed indirizzi) dei "fondatori" dei club, si nascondevano poco chiare volontà commerciali tese, più che altro, alla richiesta di denaro per l'iscrizione e per la vendita per corrispondenza di programmi più o meno originali.

Se pertanto avete intenzione di aprire un club senza alcuna finalità di lucro telefonateci il venerdì pomeriggio per esaminare le vostre proposte.

Programmi che non girano

☐ **Alcuni listati pubblicati sembrano non girare correttamente, in tutto o in parte. Cosa devo fare?** *(Aldo Guerrini - Roma. Erio Zecchini - Milano)*

• I programmi che citate non contengono errori. L'unico modo per venire fuori è quello di telefonare il venerdì pomeriggio avendo nelle vicinanze il computer su cui è caricato il programma che presenta inconvenienti.

...E il C-16?

☐ **C.C.C. tratterà argomenti inerenti il nuovo computer Commodore C-16?** *(Dario Riva - Milano)*

• Non appena il numero di richieste analogiche alla tua raggiungerà un numero sufficiente affronteremo anche il C-16.

VIC 20 INESPANSO

ASSEDIO SU MARTE

Attenzione ai missili. E' l'unica raccomandazione che vi diamo per questo semplice gioco. Da digitare sul Vic 20 (anche inespanso) e provare subito.

```
100 REM  ASSEDIO SU MARTE
110 REM  GIOCO PER VIC 20 INESPANSO
120 :
130 :
140 TI$="000000":POKE 36878,15:POKE
     36879,8
150 DIM A(22):GOSUB 710
160 Z1=7680+19*22
170 POKE 650,128
180 PRINT"[CLEAR]":Z=7680
185 PRINT"[HOME][10 DOWN][RVS]┌┌┌┌┌
    ┌┌┌┌┌┌┌┌┌┌┌┌┌┌┌┌┌[RVOFF]"
190 FOR I=1 TO 22:A(I)=1:POKE 7680+
    22*22+I-1,121:NEXT
200 PRINT"[HOME][RVS][BIANCO] ♥ ASS
    EDIO SU MARTE:ATTENDERE PREGO"
210 FOR Y=0 TO 505:POKE 38400+Y,INT
    (RND(1)*7)+1:NEXT
220 POKE 36879,8
225 PRINT"[HOME]
       ":REM  22 SPAZI
230 PRINT"[HOME][4 DOWN]";
235 PRINT"
     ":REM  26 SPAZI
```

CARATTERI E CODICI DEL C 64

Importante

I listati che sono presenti da questo numero di C.C.C. riportano, fra parentesi quadre, i nomi dei tasti funzione e di controllo cursore. Questo fatto è necessario per rendere più leggibili tutti i programmi.

Ecco che se troviamo: [CLEAR] dovremo premere contemporaneamente: tasti SHIFT e CLR/HOME ed apparirà sul vostro schermo proprio il primo simbolo della tabellina proposta.

Simbolo	Tasti utilizzati	Denominaz.	Simbolo	Tasti utilizzati	Denominaz.	Simbolo	Tasti utilizzati	Denominaz.	Simbolo	Tasti utilizzati	Denominaz.
	Shift Clr/Home	Clear		Ctrl 1	Nero		Commodore 1	Aranc		F1	F1
	Clr/Home	Home		Ctrl 2	Bianco		Commodore 2	Marr		Shift F1	F2
	Shift Crsr (S)	Up		Ctrl 3	Rosso		Commodore 3	Rosa		F3	F3
	Crsr (S)	Down		Ctrl 4	Azzur		Commodore 4	Grigio 1		Shift F3	F4
	Shift Crsr (D)	Left		Ctrl 5	Viola		Commodore 5	Grigio 2		F5	F5
	Crsr (D)	Right		Ctrl 6	Verde		Commodore 6	Verde 2		Shift F5	F6
	Ctrl 9	Rvs		Ctrl 7	Bleu		Commodore 7	Celeste		F7	F7
	Ctrl 0	Rvoff		Ctrl 8	Giallo		Commodore 8	Grigio 3		Shift F7	F8

```
240 G=PEEK(197):RT=0
250 IF RND(1)<.3 THEN 290
260 D=INT(RND(1)*22):RT=RT+1:IF RT=
    4 THEN 290:IF A(D)=0 THEN 260
270 A(D)=0:R=R+1:IF R=15 THEN Z1=Z1
    -22:FOR I=1 TO 12:A(I)=1:NEXT:R
    =0
280 POKE Z1+D,88
290 IF TI$>"000130" THEN 520
300 IF G=29 THEN X=X-1
310 POKE 36877,0
320 IF R/2=INT(R/2) THEN POKE 36877
    ,220
330 IF L>0 THEN 580
340 IF RND(1)>.9 THEN 580
350 IF G=37 THEN X=X+1
360 IF X<0 THEN X=0
370 IF X>21 THEN X=21
380 POKE Z+X1,32:X1=X
390 POKE Z+X,81
400 IF G=32 THEN 440
410 PRINT"[HOME][21 DOWN]
               [UP]  PUNTI";
420 PRINT SC"TEMPO "RIGHT$(TI$,3)
430 GOTO 240
440 FOR I=1 TO 19
450 POKE 36876,244-I*4
460 IF PEEK(Z+X+I*22)=88 THEN SC=SC
    +2
470 IF PEEK(Z+X+I*22)=65 THEN GOSUB
     990
480 POKE Z+X+I*22,93
490 IF I=>1 THEN POKE Z+X+I*22,32
500 NEXT:POKE 36876,0
510 POKE Z+19*22+X,32:GOTO 410
520 POKE 36877,0:FOR R=1 TO 10:PRIN
    T"[CLEAR][VIOLA][4 DOWN]IL VS.P
    UNTEGGIO:"SC
530 FOR U=1 TO 180:NEXT
540 PRINT"[CLEAR][GIALLO][4 DOWN]IL
     VS.PUNTEGGIO:"SC
550 FOR U=1 TO 180:NEXT
560 NEXT
570 PRINTCHR$(142):FOR U=1 TO 100:N
    EXT:POKE 197,64: RUN
580 IF L=0 THEN Q=X
590 POKE 36875,128+2*L
600 IF PEEK(Z+22*20-L*22+Q)=81 THEN
     670
610 IF L>1 THEN POKE Z+22*23-L*22+Q
    ,32
620 POKE Z+22*19-L*22+Q,65
630 IF L>18 THEN L=0:POKE Z+Q,32:PO
    KE 36875,0:GOTO 350
640 POKE 36875,0
650 L=L+4
660 GOTO 350
670 POKE Z+Q,42:POKE 36877,220:POKE
     36879,0
680 FOR T=15 TO 1 STEP -1
690 POKE 36878,T:POKE 36879,8
700 FOR M=1 TO 80:NEXT:POKE 36878,1
    5:POKE 36877,0:SC=SC-20:GOTO 61
    0
710 PRINT"[CLEAR][10 DOWN][VERDE]
    [RVS]VOLETE ISTRUZIONI?[RVOFF]
720 GET A$:IF A$="" THEN 720
730 IF A$="N" THEN PRINTCHR$(142):G
    OTO 160
740 PRINTCHR$(142)
750 IF A$<>"S" THEN 720
760 POKE 36879,8
```

CHARTBUSTERS!

1. **Ghostbusters** *(Activision)*
2. **Daley Thompson's Decathlon** *(Ocean)*
3. **Jet Set Willy** *(Software Projects)*
4. **Manic Miner** *(Software Projects)*
5. **Hunchback** *(Ocean)*
6. **Football Manager** *(Addictive Games)*
7. **Beach Head** *(US Gold)*
8. **Fighter Pilot** *(Digital)*
9. **Match Point** *(Psion)*
10. **Flight Path 737** *(Anirog)*

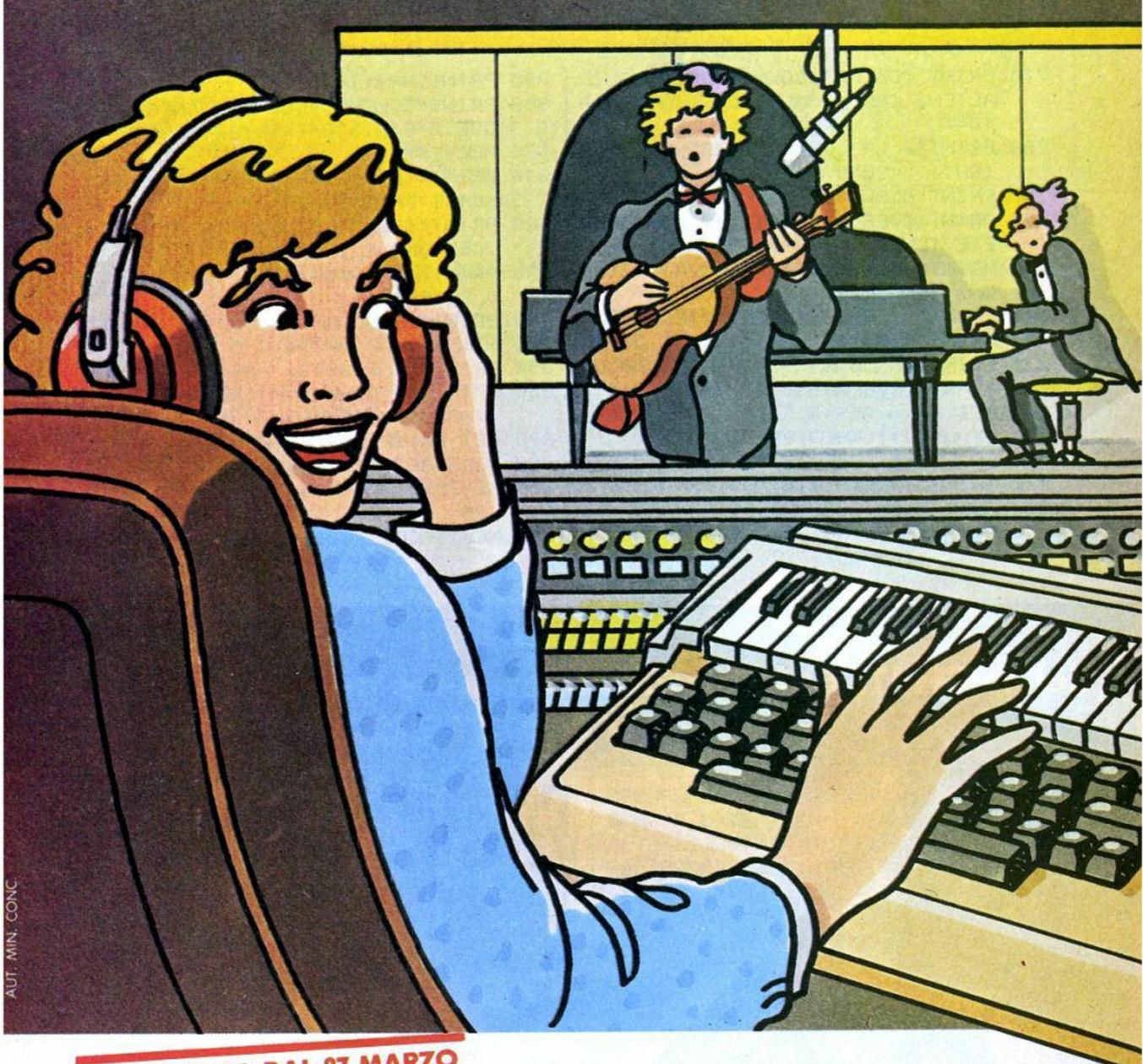
Ghostbusters l'acchiappafantasmi, si trasforma nella classifica dei videogiochi in chartbusters: acchiappaclassifica.

Dietro al game di David Crane, nella maggior parte troviamo vecchie conoscenze, almeno per i più affezionati videogiocatori.

La classifica è tratta da Microscope.

```
770 PRINT"[CLEAR][DOWN][BIANCO]GLI
    ALIENI CHE HANNO  [DOWN]ATTACCA
    TO";
780 PRINT"  LA  TERRA   [DOWN][12 R
    IGHT]  SONO ";
790 PRINT"BLOCCATI SU MARTE."
800 PRINT"[VERDE][HOME][10 DOWN]DOV
    ETE TENERLI A BADA[DOWN]";
810 PRINT" FINCHE'NON ARRIVANO I[DO
    WN] RINFORZI."
820 PRINT"[2 DOWN][BLEU]ATTENZIONE
    AI MISSILI!"
830 PRINT"[2 DOWN]     [RVS]PREMI UN
     TASTO[RVOFF]"
840 GET A$
850 IF A$<>"" THEN 890
860 FD=FD+FT:IF FD>9 THEN FT=-1
870 IF FD<1 THEN FT=1
880 POKE 7680+5*22+FD-FT,32:POKE 76
    80+5*22+FD,88:GOTO 840
890 PRINTCHR$(142)
900 PRINT"[CLEAR][DOWN][BIANCO]<-MU
    OVE A SINISTRA[4 RIGHT][DOWN]>-
    MUOVE";
910 PRINT" A DESTRA[6 RIGHT][DOWN]'
    SPACE BAR'-FUOCO"
920 PRINT"[HOME][8 DOWN]'●'-E'LA VS
    .ASTRONAVE [DOWN]";
930 PRINT"I RINFORZI ARRIVANO IN 13
    0 SECONDI"
940 PRINT"[2 DOWN][5 RIGHT]♣    2 PU
    NTI","[5 RIGHT]♠    5 PUNTI",
950 PRINT"[5 RIGHT]* -20 PUNTI"
960 PRINT"[4 DOWN]     [RVS]PREMI UN
     TASTO[RVOFF]"
970 GET A$:IF A$="" THEN 970
980 POKE 36879,8:GOTO 160
990 SC=SC+5:POKE 36879,0:FOR P=1 TO
     80
1000 NEXT:POKE 36879,8:RETURN
```

7 NOTE BIT: OGGI LA M

AUT. MIN. CONC.

È IN EDICOLA DAL 27 MARZO

LA TASTIERA COMPRESA NEL PRIMO FASCICOLO

Il 1985 è l'anno mondiale della musica… preparati con 7 note Bit, la nuova, fantastica opera che in soli 15 fascicoli quattordicinali (ciascuno corredato da una cassetta software per Commodore 64) ti insegna veramente la musica.

Con 7 note Bit, porti in casa il tuo maestro personale di musica che ti introduce alla teoria e al lessico musicali, ti insegna e ti fa fare pratica sulla tastiera, seguendoti passo passo nelle lezioni … senza mai annoiarti.

E puoi suonare subito, con l'aiuto del software "interattivo" della cassetta e della speciale tastiera musicale per il tuo 64.

7 note Bit trasforma il Commodore 64 in uno strumento musicale sofisticato ed entusiasmante.

USICA SI IMPARA COSÌ.

Con 7 note Bit vinci 30 straordinari Commodore Plus/4

Il video del computer ti presenta tutto ciò che ti serve: un metronomo elettronico, il rigo musicale, la tastiera visualizzata con le note corrispondenti ai tasti ... e potrai partecipare "dal vivo" alle session dei musicisti più famosi, suonando insieme a loro.

Prenota in edicola 7 note Bit: una prestigiosa iniziativa editoriale nata dall'unione della esperienza informatico-divulgativa del Gruppo Editoriale Jackson con la competenza tecnico-musicale SIEL. SIEL

7 note Bit è il "programma musicale" più interessante che c'è: non perdere il primo numero.

Pensa, compresa nel prezzo c'è anche la stupenda tastiera professionale per il tuo Commodore 64.

GRUPPO EDITORIALE JACKSON

D.M. 4/275256 del 2-3-85

Non tutti i leoni sono veramente Leoni.

Ecco come riconoscere un vero programma Leoni Informatica

COMMODORE 64 SOFTWARE

Leoni informatica

Garanzia a vita
Anche dopo vent'anni un programma che rivelasse un difetto d'origine viene subito sostituito.

Assistenza telefonica
Una centralina telefonica risponde ad ogni chiamata. Leoni Informatica ti fornisce anche questo servizio assicurandoti tutte le informazioni che ti necessitano per la perfetta efficienza del tuo sistema.

Aggiornamento continuo
Il mondo si evolve e le necessità cambiano. Solo per questo anche i nostri programmi possono invecchiare. Noi te li sostituiamo aggiornati.

Quando per il tuo home-computer il negoziante ti offre un programma a basso costo, diffidane. Nella quasi totalità dei casi si tratta di una copia duplicata, che per di più può non girare bene. Le conseguenze, specialmente se si tratta della tua contabilità, sono facilmente immaginabili. Leoni Informatica, Azienda leader, fa programmi da sempre, e da sempre è sinonimo di altissima qualità. Riconoscere questi programmi è facile. La classica confezione bianca e blu è accuratamente sigillata. All'interno, allegate al floppy disk, vi sono le istruzioni in italiano e, cosa importantissima, la cartolina di garanzia. Inoltre Leoni Informatica è stata la prima in Italia ad offrire la garanzia a vita, l'assistenza ed il continuo aggiornamento dei suoi programmi. Al tuo negoziante chiedi quindi la qualità, l'assistenza e l'aggiornamento: chiedi i programmi Leoni Informatica.

Richiedi a Leoni Informatica l'elenco guida ai suoi programmi.

BES Milano

Leoni Informatica non ti abbandona mai

Leoni informatica S.r.l. - Sviluppo Software - Vendita Hardware
Via Valsolda, 21 - 20143 Milano - Tel. 02-8467378-8465072

Potete trovare i nostri programmi in tutta Italia nei punti vendita Buffetti e dai rivenditori autorizzati che espongono il nostro marchio.

I GIOCHI PUBBLICATI SU COMMODORE COMPUTER CLUB

I nuovi lettori, come del resto quelli affezionati alla nostra rivista da molto tempo, troveranno di certo utile l'elenco che segue.

Rappresenta infatti un indice di rapida consultazione per rintracciare immediatamente il numero del fascicolo di C.C.C. il titolo del gioco o dell'articolo che riguarda la programmazione dei giochi, il computer su cui "gira" ed una breve descrizione del contenuto dell'articolo.

Nel caso dei giochi per il Vic 20, il termine INES significa che non è richiesta espansione di memoria. Alcuni articoli, che a prima vista potrebbero esser sc ambiati per articoli di didattica o tecnici, terminano comunque con un gioco che sicuramente sarà apprezzato soprattutto da coloro che intendono programmare giochi per proprio conto.

I lettori interessati all'acquisto per corrispondenza dei fascicoli arretrati sono pregati di tener conto di quanto segue:

- Non si effettuano spedizioni in contrassegno.

- E' indispensabile compilare la scheda riportata nell'ultima pagina di questo numero.
- I numeri 1, 2 e 7 sono esauriti ed è quindi inutile richiederli.

- Ogni tre fascicoli arretrati richiesti si ha diritto a riceverne uno gratis, che è necessario indicare con chiarezza nell'apposito modulo. E' possibile indicare, quali omaggi desiderati, anche numeri arretrati della rivista Commodore

1

- **Dadi - VIC 20 Ines**
Un semplice gioco che simula il lancio di dadi.

- **Tiro al grago - VIC 20**
Gioco di abilità.

- **Invasione spaziale - VIC 20**
Divertente giochino del tipo tiro al bersaglio.

- **Caccia al rinoceronte - VIC 20**
Attraversa la foresta evitando i rinoceronti che la abitano.

2

- *Rinoceronte col joystick - VIC 20*

I GIOCHI PUBBLICATI SU COMMODORE COMPUTER CLUB

Applicazione del joy al gioco pubblicato sul N. I di CCC.

● **Hig-low - VIC 20**
Alternativamente il computer e il giocatore devono indovinare un numero casuale.

● **Nim - VIC 20**
Noto gioco in cui occorre ragionare per battere il computer.

3

● **Tot 9 - VIC 20**
Uno dei programmi elaborati dallo studio Carchidi di Roma.

● **Labirinto – VIC 20**
Semplice giochino di abilità.

5

● **Il ferroviere - VIC 20**
Divertente giochino che consiste nel guidare con accortezza una serie di convogli ferroviari.

6

● **Atterraggio - VIC 20 INES**
Gioco di abilità consistente nel guidare una piccola astronave attraverso uno schermo pieno di meteoriti.

Roulette russa - VIC 20 INES
In una pistola c'è un solo proiettile. A chi è destinato?

● **Antiaerea - VIC 20**
Mini gioco per il popolare computer.

● **Forza 4 - SISTEMI & VIC 20 INES**
Simulazione col computer del notissimo gioco.

7

● **Apprendimento automatico**
Interessante applicazione che illustra un metodo per risolvere un indovinello.

● **Controllo del cursore e videogiochi - VIC 20**
Come controllare lo spostamento del cursore sullo schermo mediante la pressione di alcuni tasti.

Continua nel prossimo numero

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... **Cognome** ....

Via .... **N°** .... **CAP.** .... **Città** ....

Telefono .... **Orario** ....

Registrate il mio abbonamento annuale a Commodore Computer Club.

☐ Ho versato oggi stesso il canone di L. 28.000 a mezzo c/c postale n° 37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

☐ Ho inviato oggi stesso assegno bancario n..
per l'importo di L. 28.000 intestato a Commodore Computer Club.
Si prega di scrivere il proprio nome e l'indirizzo completo in modo chiaro e leggibile. Inviare la fotocopia del bollettino di c/c postale.

Considerando che i numeri 1, 2 e 7 sono esauriti, vogliate inviarmi i numeri arretrati .... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri. o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L............... Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L.................. a mezzo c/c postale n. 37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

STATISTICA

Non posseggo un computer ☐

Posseggo un C64	si ☐	... no ☐
Posseggo un VIC 20	si ☐	... no ☐
Posseggo un Commodore Plus 14	si ☐	... no ☐
Posseggo un Commodore Plus 16	si ☐	... no ☐
Posseggo un registratore dedicato	si ☐	... no ☐
Posseggo un drive 1541	si ☐	... no ☐
Posseggo una stampante	si ☐	... no ☐
Posseggo un monitor	si ☐	... no ☐

COLLABORAZIONE

A titolo di prova vi invio un articolo e la cassetta..........................disco....................................
col programma che intendo proporre per la pubblicazione di cui garantisco l'originalità.

DOMANDA/RISPOSTA ....

....

....

....

....

....

RICHIESTA ARGOMENTI

Mi farebbe piacere che Commodore Computer Club parlasse più spesso dei seguenti argomenti:

1/ ....

2/ ....

3/ ....

4/ ....

GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .... Voto ....

B/ .... Voto ....

C/ .... Voto ....

D/ .... Voto ....

PICCOLI ANNUNCI ....

....

....

....

....

....

CERCO/OFFRO CONSULENZA ....

....

....

....

....

....

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... **Cognome** ....

Via .... **N°** .... **CAP.** .... **Città** ....

Telefono .... **Orario** ....

In edicola questo mese